十日学会易语言

图解教程

主编：王军

编辑 :张志恒　马展保　熊立安

校对：史世恒

（2011年重新整理版）

电话 :86-0411-88995831 传真 :86-0411-88995834

内容简介

本书用图解的方式对易语言的使用方法和操作技巧作了生动、系统的讲解。全书分十章，分十天讲完。

第一章是介绍易语言的安装，以及运行后的界面。同时介绍一个非常简单的小程序，以帮助用户入门学习。最后介绍编程的输入方法，以及一些初学者会遇到的常见问题。第二章将接触一些具体的问题，如怎样编写一个 1+2等于几的程序，并了解变量的概念，变量的有效范围，数据类型等知识。其后，您将跟着本书，编写一个自己的 MP3播放器，认识窗口、按钮、编辑框三个常用组件。以认识命令及事件子程序。第三章章主要介绍易语言的命令概念，并举出一个大小数判断的例子 ,介绍判断语句，以及介绍选择语句和循环语句。第四章介绍常数、常量、资源的应用。第五章主要介绍应用程序菜单的制作，并举出一个记事本的例子，介绍判断语句，以及介绍选择语句和循环语句。第六章学习静态变量、变量数组及动态管理变量。第七章介绍组件的应用，并用几个简单的小例程来了解组件的属性，事件，和方法。第八章主要介绍“ 易语言 ”子程序的调用方法、子程序参数的使用方法以及参数属性的相关使用方法。第九章主要介绍“ 易模块 ”的安装、使用方法以及新建、保存的方法。同时介绍一个非常简单的“ 易模块 ”编写过程，以帮助用户了解和学习。第十章简单介绍 API的应用。

另外，本书在每章后面都附有课后练习题，以帮助读者巩固所学的知识。本书内容丰富、由浅入深、通俗易懂、图文并茂、范例丰富、讲练结合，编者力求在实例的演示中教会读者真正掌握易语言的基本技能和操作方法。本书不但是针对小学六年级以上的入门者最佳的自学指导书，同时也是国内各种职业技术学校和社会电脑初级培训班的首选教材。本光盘内容为本版电子书。

目　录

第 1 章 “易语言”基础知识

本章主要介绍“易语言”的下载安装，以及运行后的界面。同时介绍一个非常简单的小程序，以帮助用户入门学习。最后介绍编程的输入方法，以及一些初学者会遇到的常见问题。

本章学习内容：

注意：安装与运行易语言对于电脑的基本要求：

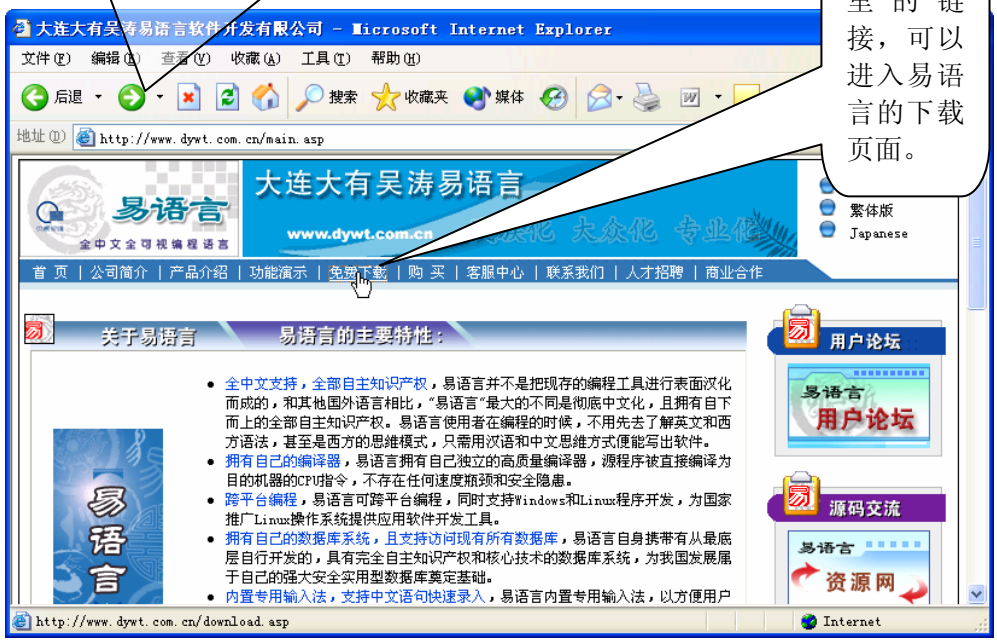
“易语言”对硬件要求不高，只要可能运行 Win 95 以上的机器都可以使用易语言。“易语言”运行的最低计算机配置要求为：

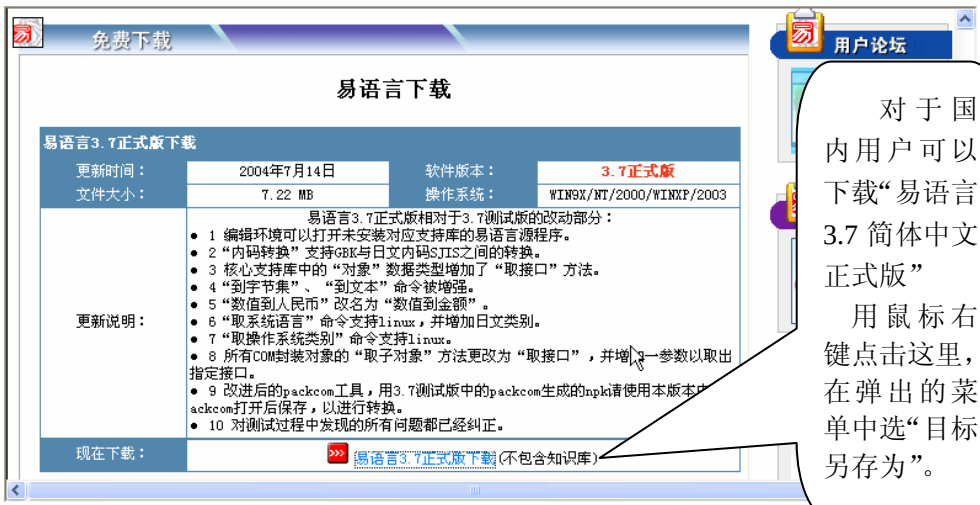
- Windows 9X / 2000 / NT/XP/2003 操作系统；
- 推荐 Pentium 或更高的处理器；
- VGA 或分辨率更高的显示器；
- 16MB 内存（推荐 128MB 以上内存）；
- 鼠标或其他定点设备。
- 若可上英特网，可直接下载易语言最新版本，且从论坛上得到大量例程等。

1.1 如何下载易语言

如何从网上下载最新版本的“易语言”呢？大家可以在浏览器中输入以下的网址：www.eyuyan.com

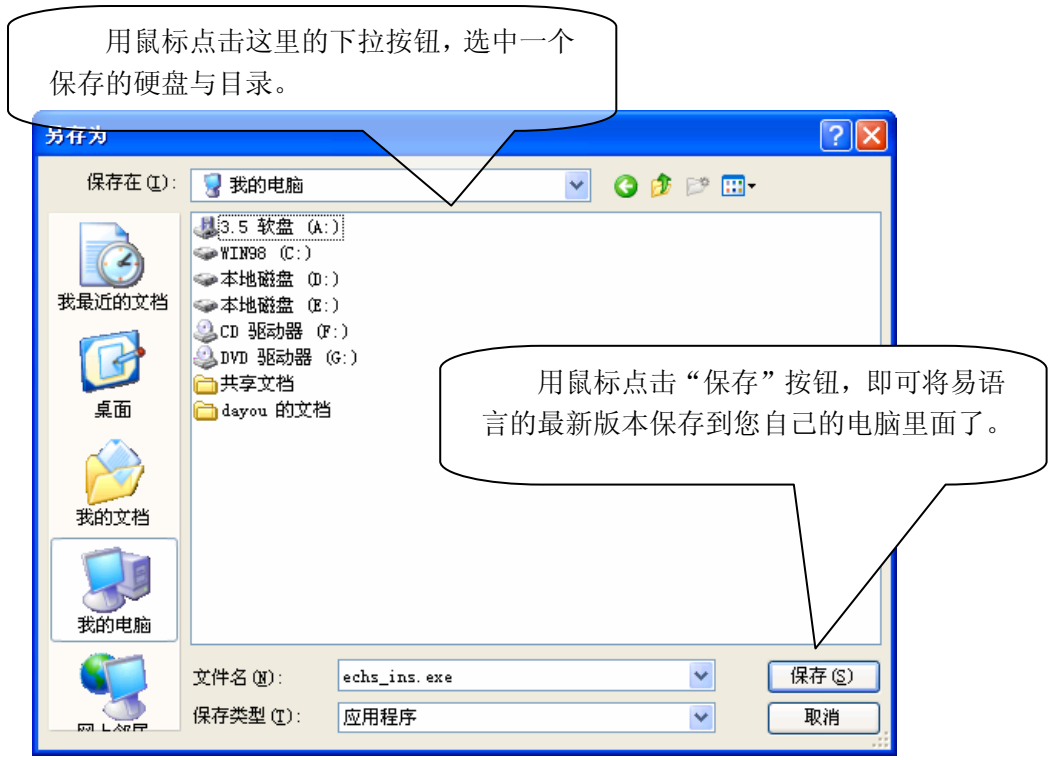
用鼠标点击这里的链接，可以进入易语言的下载页面。





对于国内用户可以下载“易语言3.7 简体中文正式版”

用鼠标右键点击这里，在弹出的菜单中选“目标另存为”。



下载页面中还可以下载以下的程序，它们的作用分别是：

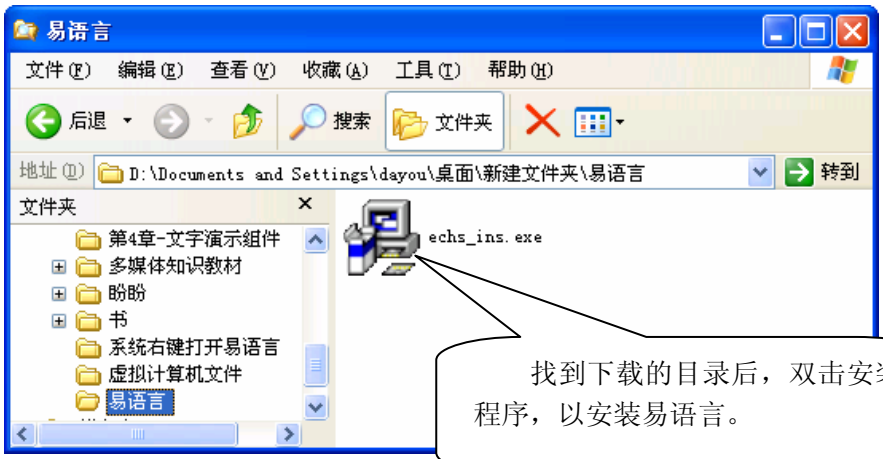
加密狗驱动安装程序：如果您已注册了易语言加密狗企业版，则必须首先下载并安装此驱动程序。

《易语言零起点》教程：Liigo 编写的供初学者学习的易语言教程，适合于初学者向中级进阶学习。

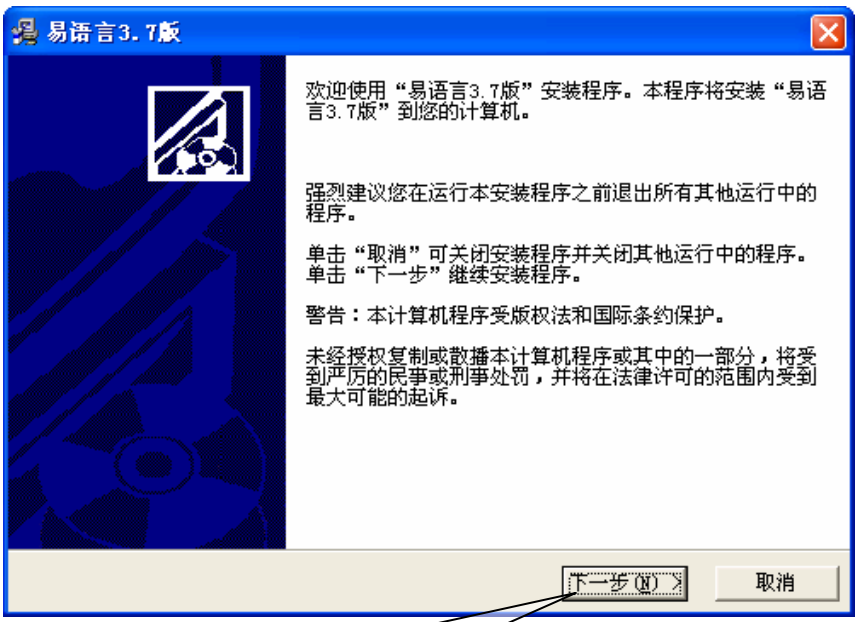
易语言 2.53 简体中文免费版：此为以前的易语言版本，没有加入任何限制，现提供给大家作为了解易语言用。可生成 EXE 可执行文件。

Win95 正常使用补丁：在 Windows95 系统下运行易语言或易程序前必须首先安装此补丁。

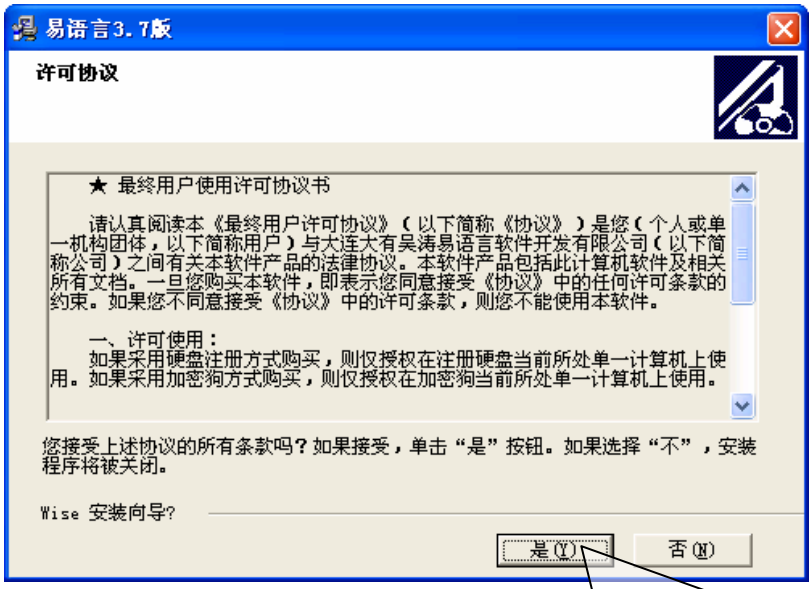
1.2 如何安装易语言



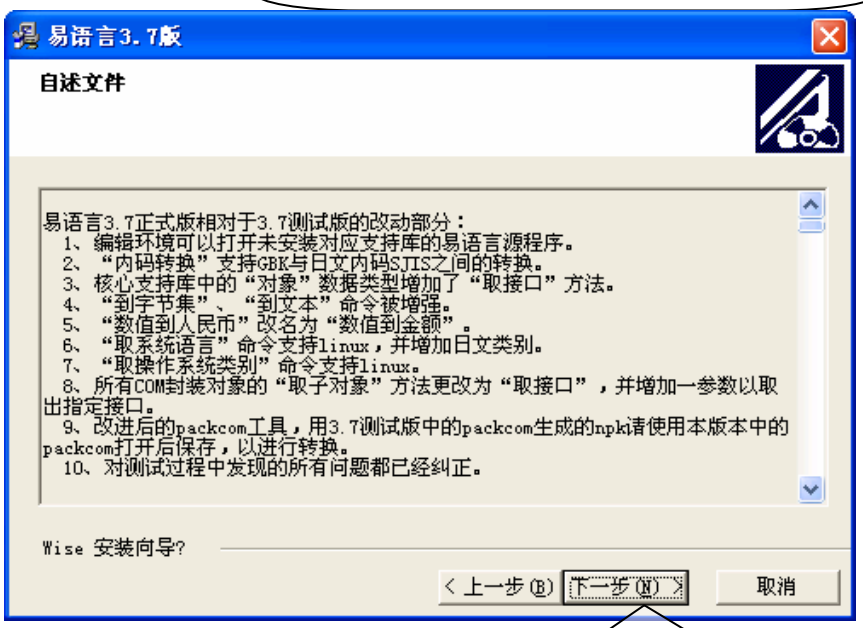
找到下载的目录后，双击安装程序，以安装易语言。



进入欢迎安装界面。
点击下一步，继续安装易语言。



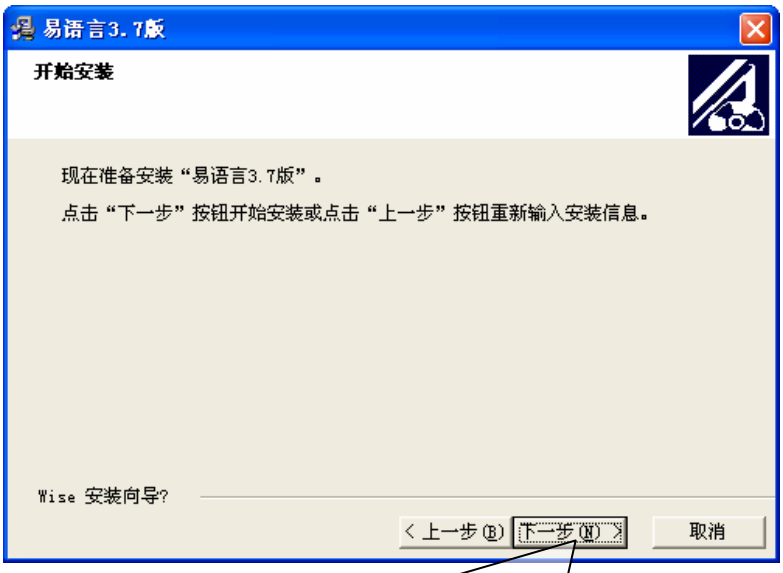
进入用户使用许可协议界面。
点击“是”钮，继续安装易语言。



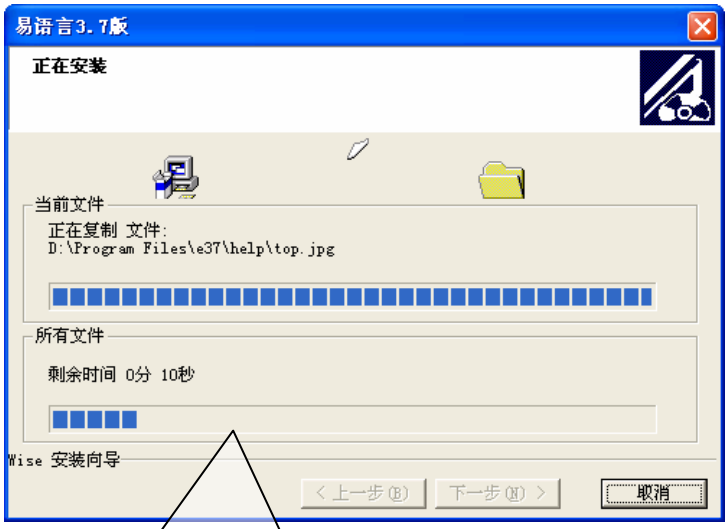
通过自述，可以了解易语言的新增功能。
点击“下一步”钮，继续安装易语言。

点击“浏览”按钮，选择其它的安装目录。

点击“下一步”钮，继续安装易语言到默认安装目录中。



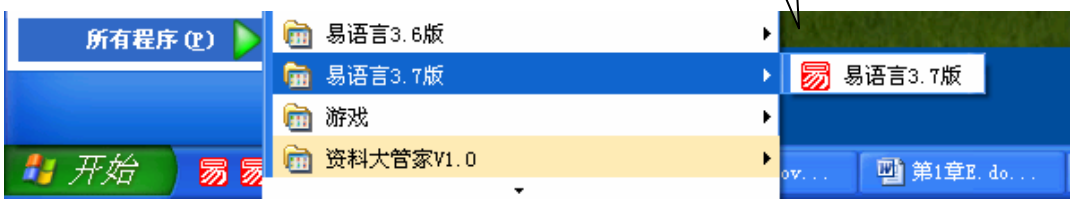
点击“下一步”按钮，正式开始安装易语言。



正在进行安装，这里是安装进度。

安装完成后，会在桌面上形成一个快捷方式，用于运行易语言。

也可以在“开始”菜单中找到运行易语言的快捷方式。



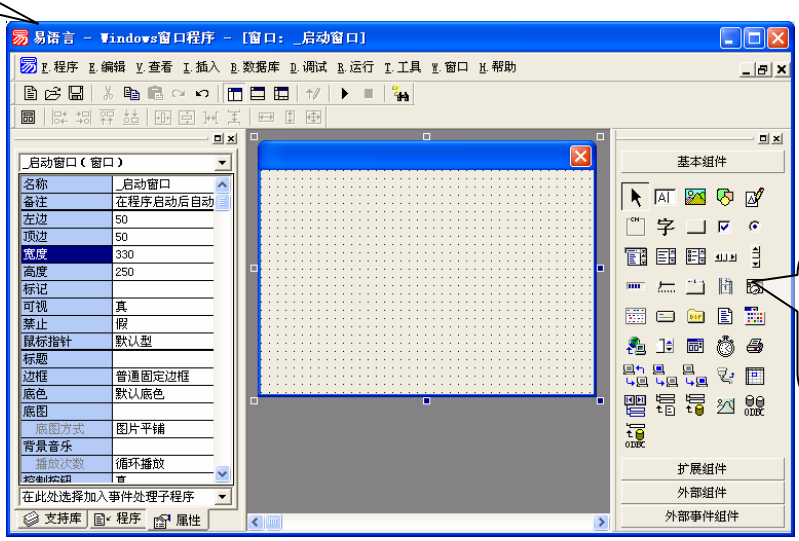
1.3 了解易语言的界面

前面已说过了如何启动易语言，只要双击易语言快捷方式即可。下面，让我们来看看易语言的界面吧。

在这里打勾，下次启动时将不显示此对话框。直接生成一个 Windows 窗口程序易程序。

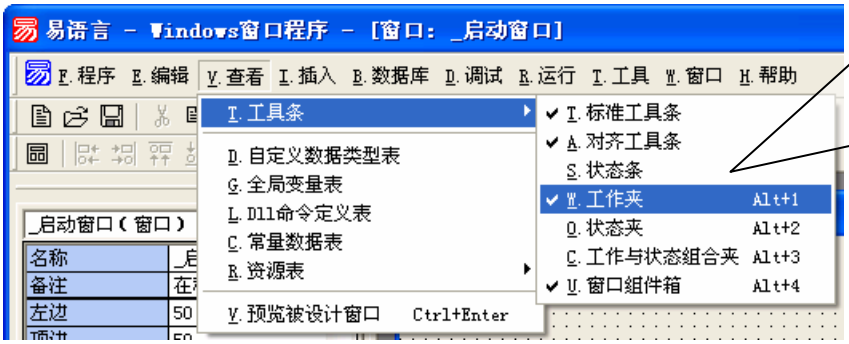
运行易语言后，首先会显示易语言的新建对话框。这里大家先选“确定”钮进入。

标题栏

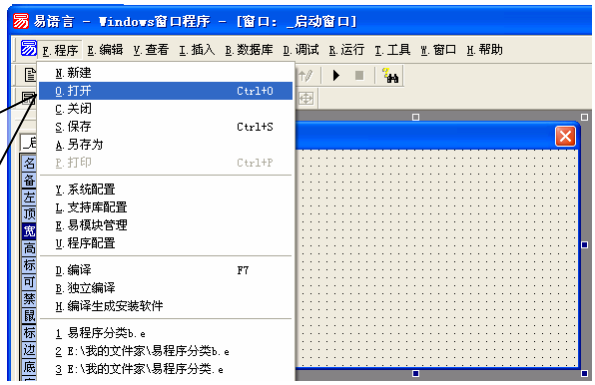


工作与状态组合夹

组件箱

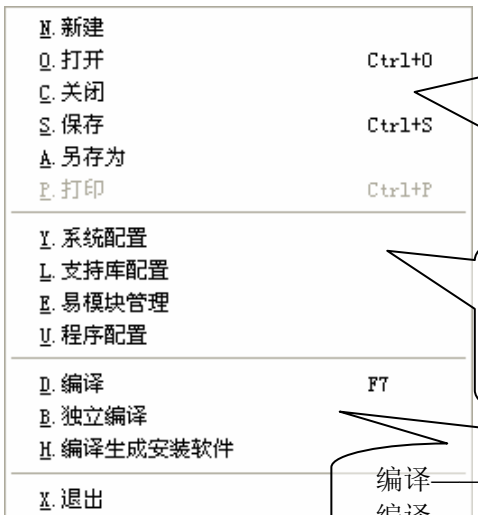


可以通过菜单“查看”→“工具条”，以显示或隐藏这些工作夹。



菜单中归类列出了易语言的功能命令，通过菜单可完成诸如打开易程序，保存易程序等功能。

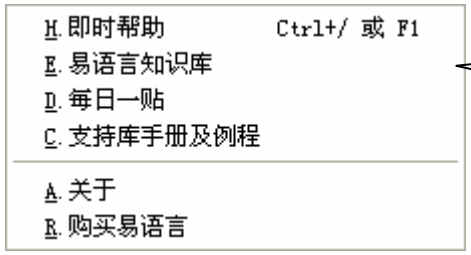
下面先介绍两个菜单吧。其它菜单后面将有专门介绍。



新建——新建一个易程序（*.e）
打开——打开一个易程序（*.e）
关闭——关闭已打开的易程序
保存——保存易程序
另存为——以另一个文件名保存
打印——打印当前程序集的程序源代码

系统配置——设置易开发环境选项
支持库配置——管理其它支持库
易模块管理——对易语言模块进行管理

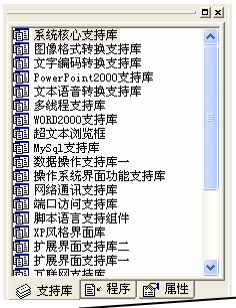
编译——生成可执行程序（*.exe）非独立发布版本
编译——生成可执行程序（*.exe）独立发布版本
编译生成安装软件——生成自动安装版本（*. exe）



随时在程序设计中按下 F1 键可得到与主题相关的帮助。

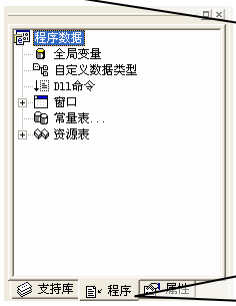
标题栏可以看到易语言的目的平台，文件名，当前窗口等信息。

工具条是一些菜单中的常用命令，用一个图标分别表示命令的含义。

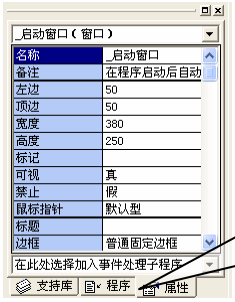


支持库面板中显示了易语言的命令分类、库定义数据类型和库定义常量。

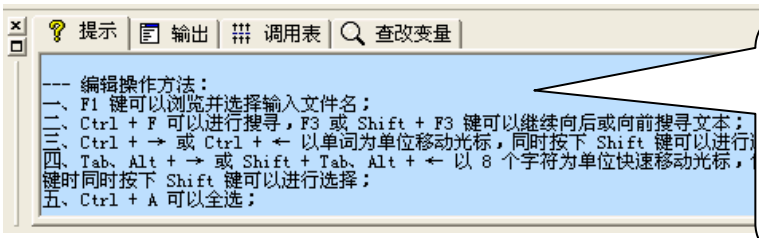
组件工具箱中列出了易语言五十多种组件，并被分类。



程序面板中显示了一个程序的资源列表。



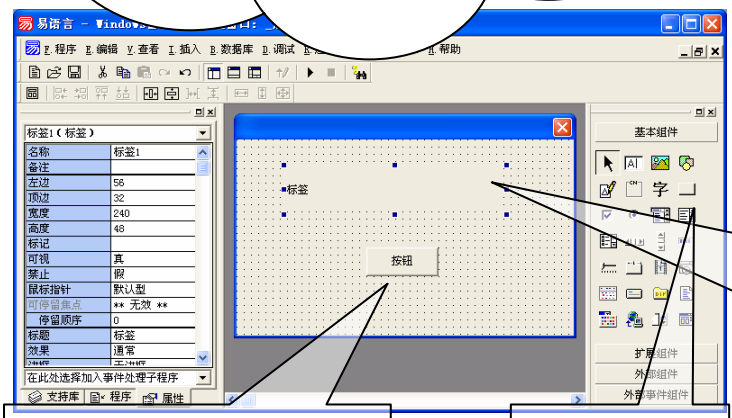
属性面板中列示了组件的属性表和组件列表，事件列表。



状态夹中显示了提示信息，还有编译时的输出信息，以及调试时的调用表信息等。

1.4 开始写第一个易程序

在 Windows 下编程已是非常简单的事情，比 DOS 下编程容易得多。因为在 DOS 下，可视化的程度不高，编程与调试是分开的，要编写一段程序后，再运行一下看看，而且编写菜单与窗口系统非常不容易。而在 Windows 下，都是所见即所得的编程手法，这样对于普通用户来说，编程就相对容易了。

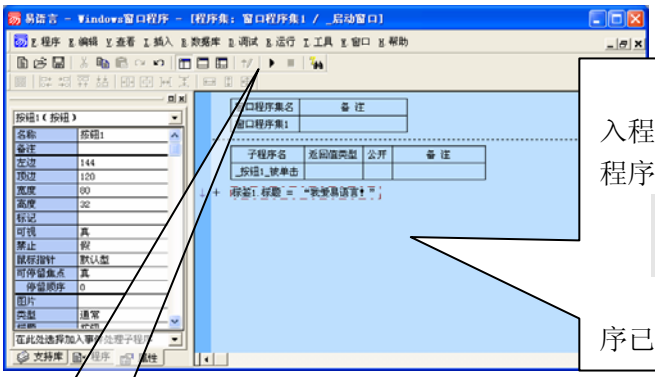


① 点击“标签”组件。

②在窗体上拉出一个标签组件。系统自己动命令为“标签 1”。（按住鼠标左键不松手，拖放到右下角）

③ 点击“按钮”组件。

④在窗体上拉出一个按钮组件。系统自己动命令为“按钮 1”。下面再双击它。

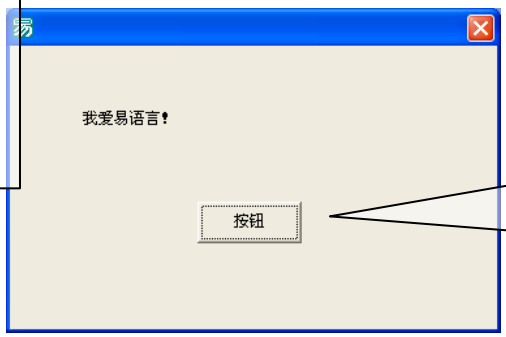


⑤双击按钮组件后，进入程序设计界面，输入此行程序代码：

```
标签 1.标题 = “我爱易语言！”
```

恭喜你，你的第一个程序已经建立。

⑥点击试运行按钮，试运行程序。或按 F5 键也能试着运行这个程序。

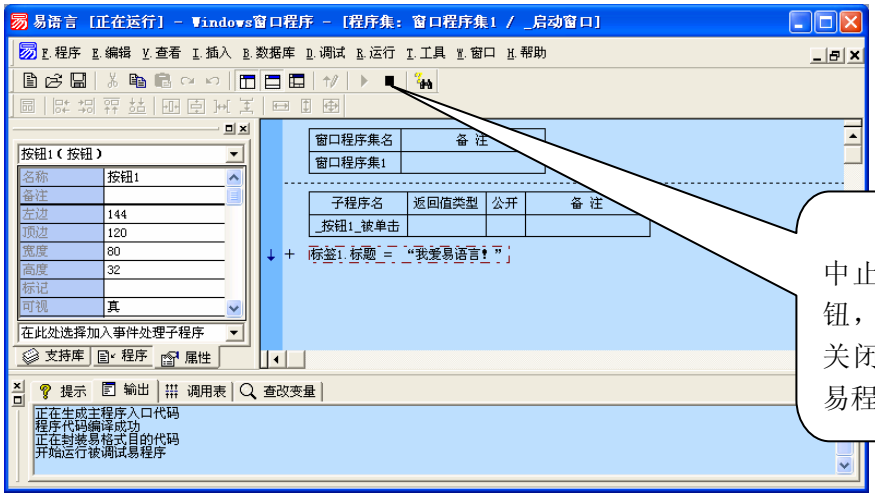


这是运行后的效果。可以点击按钮，看到标签的标题有所改变。

下面来分析为什么此程序能够完成这样的功能。

首先需要了解的是启动窗口是所有程序的平台，所有的内容都要显示在上面，因此一个程序不能没有一个主窗口，否则无法输入，也无法显示结果。在此窗口上有两个组件，一个是用来进行动作的按钮组件，当用户单击时就会控制标签显示文字，当然显示文字的过程是通过改变标签的标题属性所形成的，这样大家看上去就象是按钮在控制标签，让标签显示文字了。

关闭被运行的例程（可以单击窗口右上角的关闭按钮关闭这个小程序）



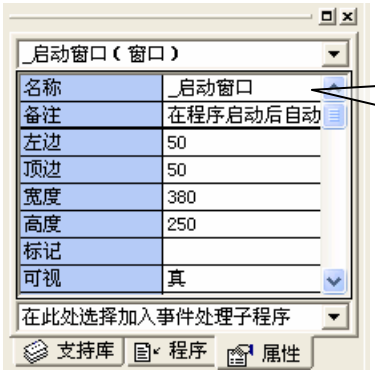
或点击中止运行按钮，也可以关闭运行的易程序。

1.5 了解编程

下面通过分析第一个易程序，来了解什么是编程。

1.“_启动窗口”的作用

“_启动窗口”的作用是非常重要的，当程序启动后自动调入本窗口。



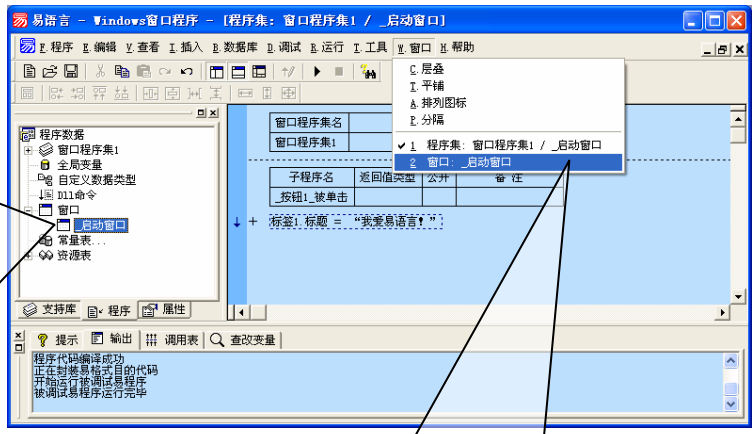
在属性表中可以看到“名称”一栏的内容为：“_启动窗口”该名称就是此已被选中的程序窗口的名称。

凡是以短下划线“_”开头的名称都是具有特定意义的名称。名称为“_启动窗口”的程序窗口，易程序在运行起来后会自动载入并显示，这就是例程执行后能够马上显示出窗口的原因。

大家也不要更改这个窗口 的名称。

注意：

如果正处于程序设计界面，没有窗口界面出来，右侧没有出现“_启动窗口”，可以用鼠标双击左边“_启动窗口”，弹出启动窗口



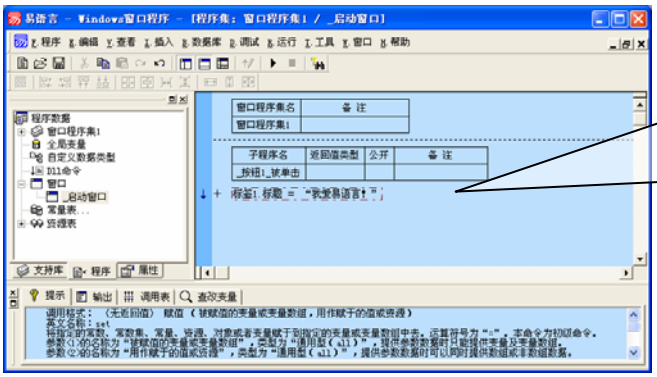
也可以通过主菜单“窗口”菜单在各打开的窗口与程序集之间切换。

2．按钮与标签的作用

下面我们分析一下按钮组件与标签在程序设计中起到了什么样的作用。

双击程序窗口中的“按钮”组件，将会跳转到对应的代码设计界面。

注意：有些控件双击会进入默认的事件子程序，实际上也可通过属性面板中的下拉菜单中找到所有的事件子程序。



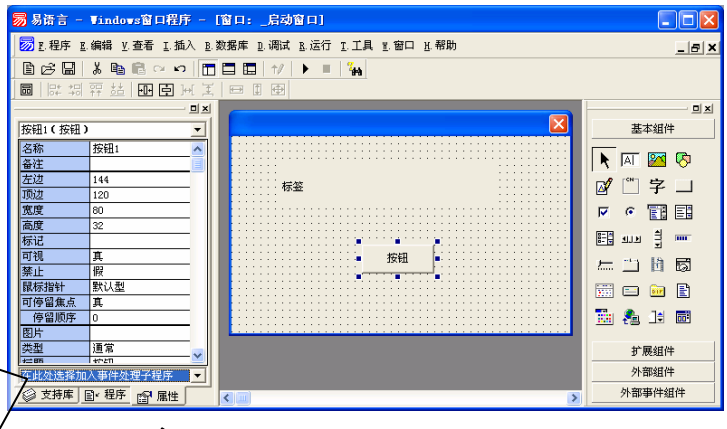
在程序设计界面中可以看到下面这条语句：

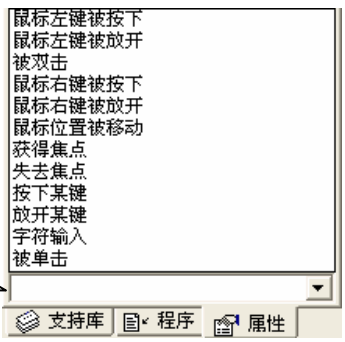
```
标签1.标题 = “我爱易语言！”
```

这条语句具体是什么意思呢？为什么它会被放在名称为“_按钮_1_被单击”的子程序中？

请跳转到程序窗口设计界面（双击左侧程序面板中的“_启动窗口”）。

单击选择窗口中的按钮，在属性表中可以看到其名称为“按钮1”，单击属性表底部显示为“在此处选择加入事件处理子程序”的组合框。



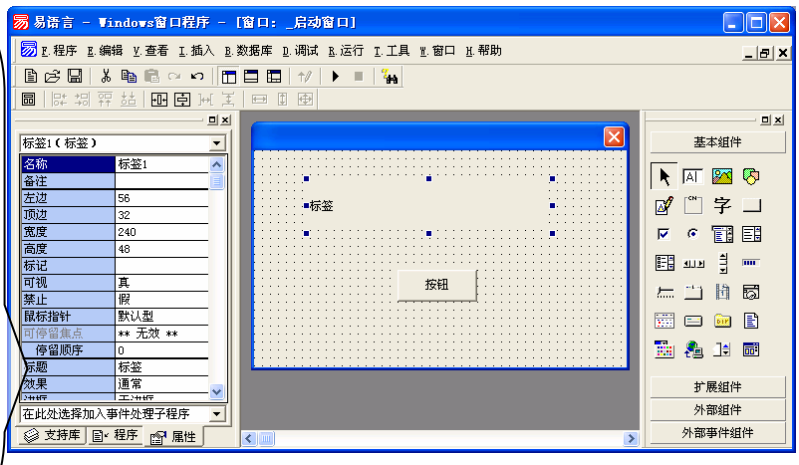


“事件处理子程序选择框”中有很多事件选择项，在其中可以找到名为“被单击”的列表项。

前面曾经提到过：凡是以短下划线“_”开头的名称都是具有特定意义的名称，此处也不例外。名称为“_按钮 1_被单击”或类似名称的子

程序被专门用作接收运行时来自程序窗口的事件，被称为事件处理子程序。它们名称的组成格式为“_”+ 产生事件的窗口单元名称 + “_” + 事件名称。按照此格式分析即可得知，名称为“_按钮 1_被单击”的子程序就是被用来接收名称为“按钮 1”的按钮窗口单元上所产生的“被单击”事件。也就是说，易程序运行时用户一旦单击了此按钮，系统将会自动执行具有此名称的子程序。

然后选中窗口中的标签控件字，在属性表中可以看到其名称为“标签 1”，并且属性表中同时还有名为“标题”的属性表栏。



引用一个窗口单元必须使用它的名称，引用窗口单元的某个属性必须使用： 窗口单元名称+“.”+属性名称 的格式，由此知道，“标签 1.标题”实际上就是引用名称为“标签 1”的窗口单元的“标题”属性，而语句：

```
标签 1.标题 = “我爱易语言！”
```

执行后就是将“标签 1”窗口单元的标题改变为“我爱易语言!”。所有在程序中使用的文本数据两边都必须用双引号括起来，譬如“我爱易语言!”。

自动载入并显示名称为“_启动窗口”的程序窗口

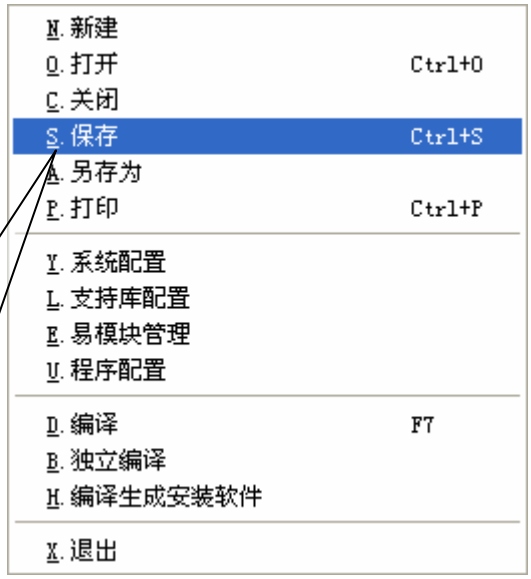
当用户按下窗口中的按钮后，“_按钮1_被单击”子程序被自动执行

程序代码：

```
标签1.标题 = “我爱易语言！”
```

语句被执行，改变了标签1的标题，从而显示出对应文本。

一个简单的编程步骤就是在“_启动窗口”上画一些控件，并且接受使用者的事件动作，用子程序改变控件的属性，或进行处理后显示出来。



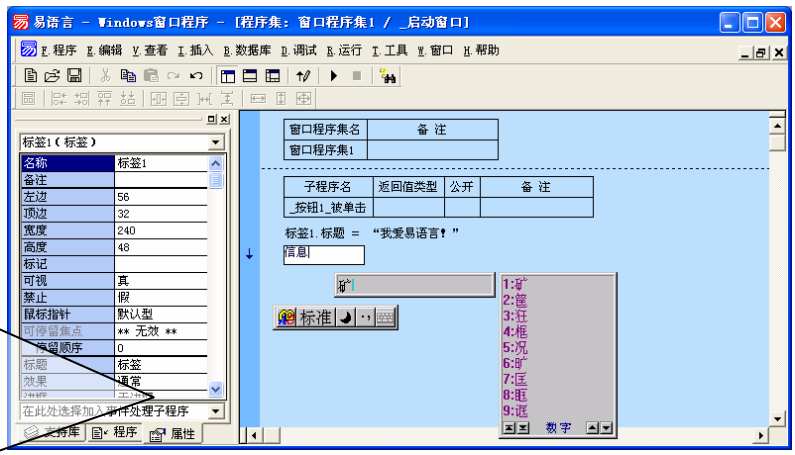
最后，别忘了保存你的易程序。

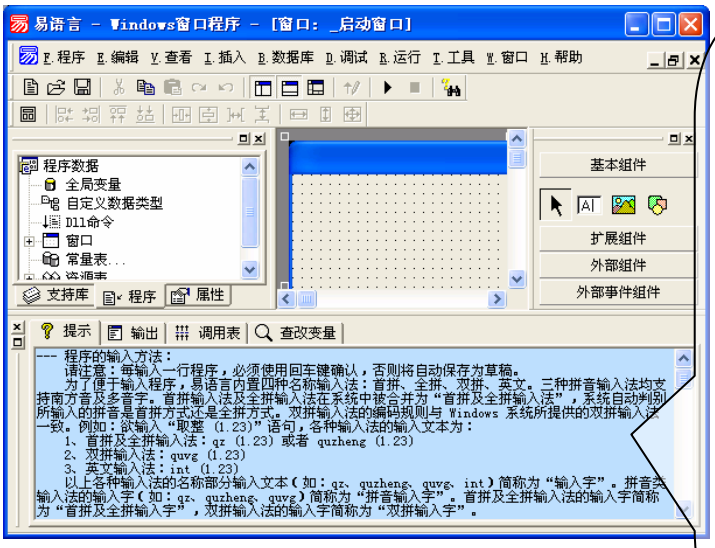
使用菜单的“程序”→“保存”，或“程序”→“另存为”命令，保存你的程序。

1.6 如何较好地输入程序代码

程序代码输入的几种方法

“易语言”可以直接使用系统汉字输入法，如五笔、全拼、智能ABC、双拼、自然码、二笔输入法等，直接输入中文程序语句。



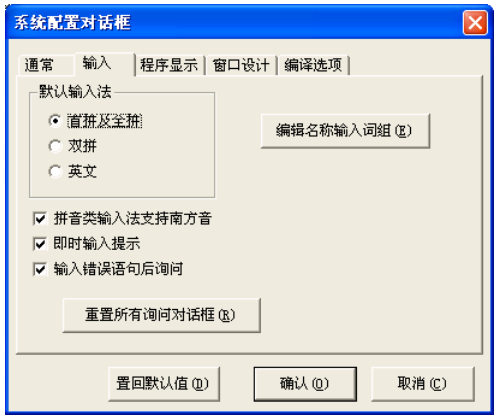


初次运行易语言时，会在提示夹中显示如何使用易语言的内置输入法。在这里其实已介绍得非常详细了。

这四种内置的输入法分别是：首拼、全拼、双拼、英文。前三种拼音输入法均支持南方音及多音字。

首拼输入法及全拼输入法在系统中被合并为“首拼及全拼输入法”，系统自动判别所输入的拼音是首拼方式还是全拼方式。

配置输入法的窗口通过单击“易语言”主菜单的“程序”→“系统配置”→“输入”得到。



双拼输入法的编码规则与 Windows 系统所提供的双拼输入法一致。例如：欲输入“取整 (1.23)”语句，各种输入法的输入文本为：

首拼及全拼输入法：qz (1.23) 或者 quzheng (1.23)

双拼输入法：quvg (1.23)

英文输入法：int (1.23)

以上各种输入法的名称部分输入文本（如：qz，quzheng，quvg，int）简称为“输入字”。拼音类输入法的输入字（如：qz，quzheng，quvg）简称为“拼音输入字”。首拼及全拼输入法的输入字简称为“首拼及全拼输入字”，双拼输入法的输入字简称为“双拼输入字”。

输入字可以用来输入程序中所涉及到的一切名称，包括：

- 所有当前运行支持库中所提供的命令、窗口和报表单元数据类型及其属性和方法、普通数据类型及其成员和方法、库定义常量等等名称；
- 用户在程序中定义的子程序参数、子程序局部容器、程序集容器、全局容器名称；
- 用户定义的子程序、 DLL 外部命令名称；
- 用户定义的数据类型及其成员名称；
- 用户所加入的资源和所定义的常量的名称；
- 用户在设计窗口或报表时所定义的窗口单元、菜单项目或报表单元名称；
- 系统数据类型名称，如：“整数型”、“小数型”等等；
- 系统常量名称，如：逻辑值常量“真”和“假”等。

在使用首拼输入字时，需要注意纯韵母发音汉字的输入。如：“按钮”中的“按”字，它的发音是韵母 an，对于此类汉字，在首拼输入法中必须写全，譬如“按钮”的首拼输入字就应该为 ann（即 an，n）。

输入字类型指定

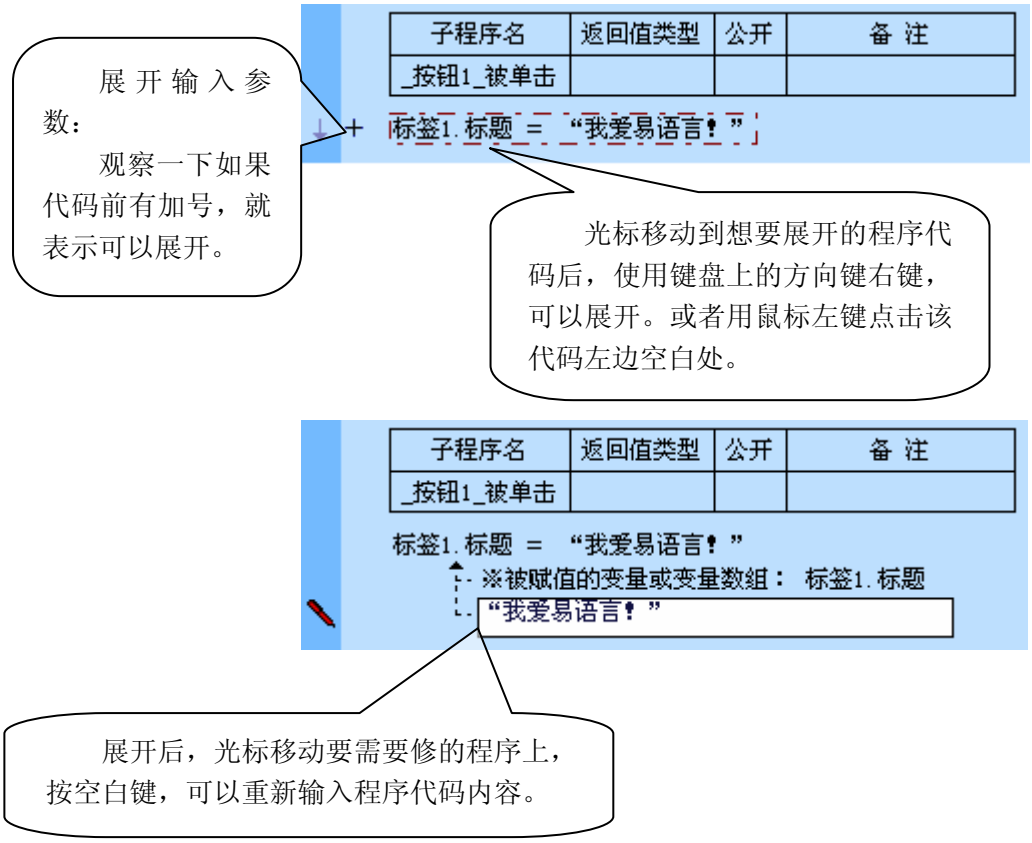
在程序中书写输入字时，可以使用一个半角符号来引导该输入字，以指定其类型。各输入字的类型引导符号如下。

首拼及全拼输入字：分号，如“；qz”，“；quzheng”。

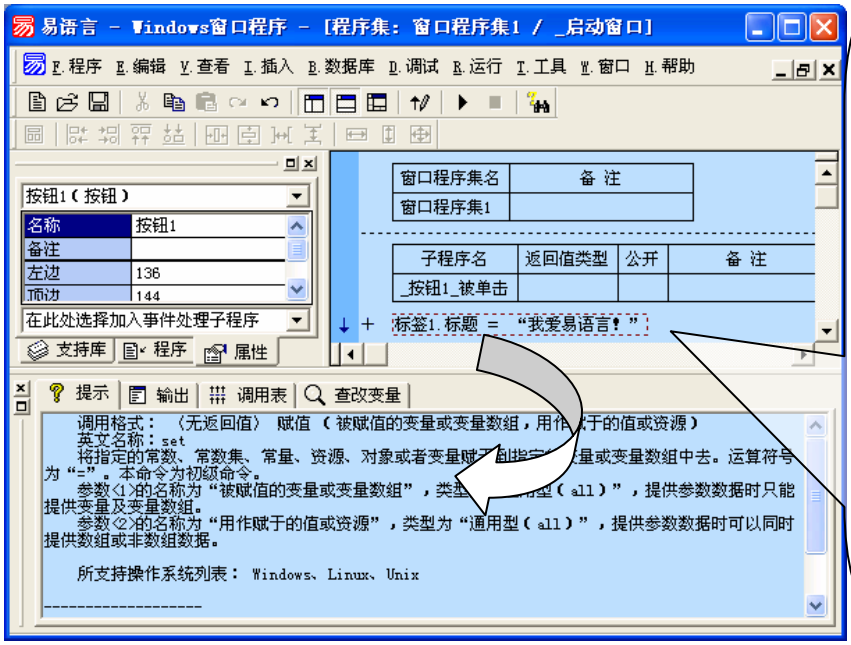
双拼输入字：冒号，如“：quvg”。

英文输入字：单引号，如“’int”。

系统具有一个当前默认输入法状态，如果某输入字前没有加上类型引导符号，则默认是属于该输入法的输入字。系统安装完毕后，当前默认输入法为“首拼及全拼输入法”。这就意味着，在当前默认输入法为“首拼及全拼输入法”的时候，要想使用双拼输入字，则必须在输入字的前面加上类型引导符“：”。不过此时英文输入字前可以加也可以不加类型引导符“ ’”。其原因为：如果当前默认类型输入字不能找到匹配的目标名称时，系统将自动把该输入字转换为英文输入字后再去寻找匹配名称一次。



1.7 “易语言”的帮助系统



光标移动到想要得到帮助的程序代码行上，按键盘上的F1功能键，立即在下方的提示夹中得到帮助提示。

1.8 初学者的常见问题

1．初始配置信息

本章及以后所有叙述都假设在安装易语言后，没有使用“程序”→“系统配置”菜单项修改过系统的初始配置信息。如果不能确定，请在启动易语言的同时一直按住 Shift 键，出现系统界面后再放开，此时将自动清除以前的设置信息。

2．输入程序后一定要回车

如果在输入一行程序后，没有按回车键，这时系统认为没有确认，所以在程序语句前面会加上“草稿”两个字。如果想去除“草稿”两个字，就要输入后加入回车确认。如果已有“草稿”两个字，想要去除，可以激活想要改的程序行。激活方法是在要改的一行处按键盘上的空白键，或用鼠标双击此程序行。

3．如何去除底图

有时在窗口属性里加了一幅图，现在想去掉，但是没有那个选项，总显示“有数据”，此时请选中该属性后直接按 Delete 键。在最新的版本中，可以用鼠标右键弹出删除的命令。

4．如何消除标签框中“标签”两字

方法：请修改标签的“标题”属性。即进入属性面板，将标题属性后的文字更改。

5．如何即时查询帮助信息

方法：如欲对系统中各运行支持库内的命令、库定义数据类型、库定义常量等等信息进行查找，请在易系统启动后使用以下方法之一：

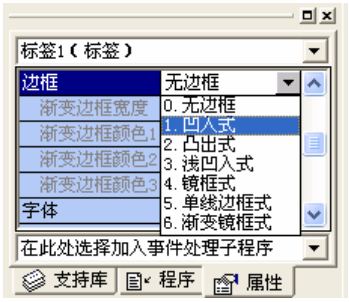
（1）直接在工作夹内的支持库栏中单击对应的所欲查找其信息的项目，此时所有的相关信息将会显示在系统的提示栏或者状态行中。

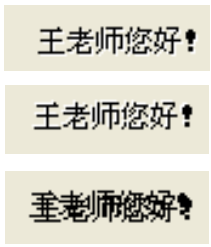
（2）如果欲将这些信息提取出来打印或者以后阅读，请在相应项目上单击鼠标右键，在所弹出的菜单中选择“拷贝帮助文本到剪贴板”或者“写帮助文本到文件”功能，输出与该项目及该项目所有子项目相关的帮助信息。譬如：在支持库名项目上进行此操作，将输出此支持库内的所有信息。

1.9 课后练习

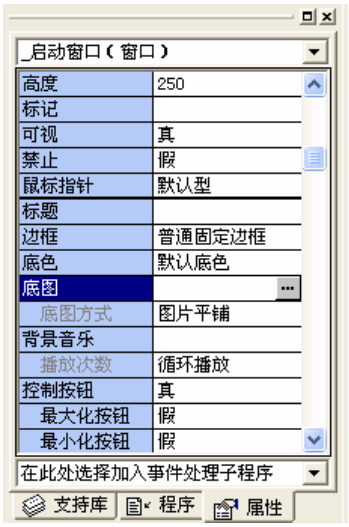
（1）将本章例子中的文字内容改为向老师打招呼的语句。

（2）练习更改标签标题的效果。



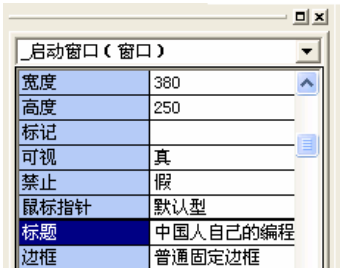


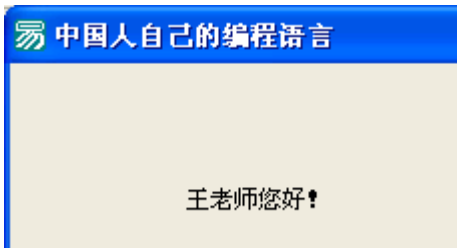
（3）在老师或有基础的朋友的指导下，将窗口填上一张图片。



更改底图显示方式。

（4）给窗口加上标题文字。





第 2 章　编程的基础概念

在本章中，将接触一些具体的问题，如怎样编写一个 1+2 等于几的程序，并了解变量的概念，变量的有效范围，数据类型等知识。

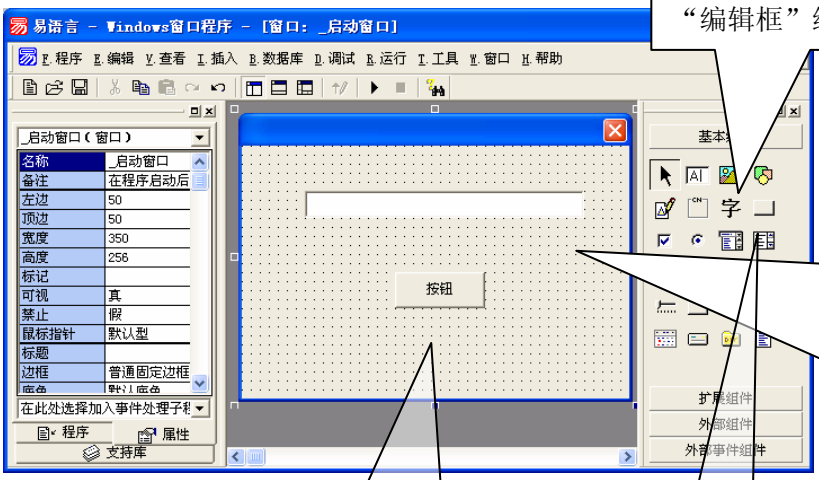
其后，您将跟着本书，编写一个自己的 MP3 播放器，认识窗口、按钮、编辑框三个常用组件。以认识命令及事件子程序。

本章学习内容：

大家会说 1+2 等于几这还不知道呀，这个例程虽然简单，也是在为以后进行复杂一些的计算作准备。我们先用最简单的例子来演示，大家只有会写这个小程序了，那么就算复杂一些的计算，都可以自己写了。

- 那么如何用易语言实现呢，大家就跟着下面的步骤来吧。

2.1 编写第二个易程序

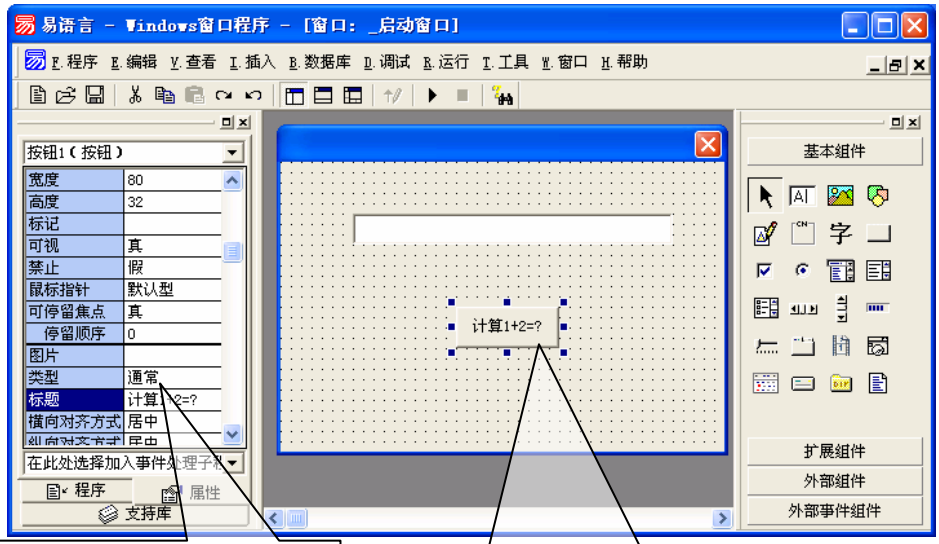


①首先启动易语言。点击“编辑框”组件。

②在窗体上拉出一个编辑框组件。形成“编辑框1”。(按住鼠标左键不松手，拖放到右下角)

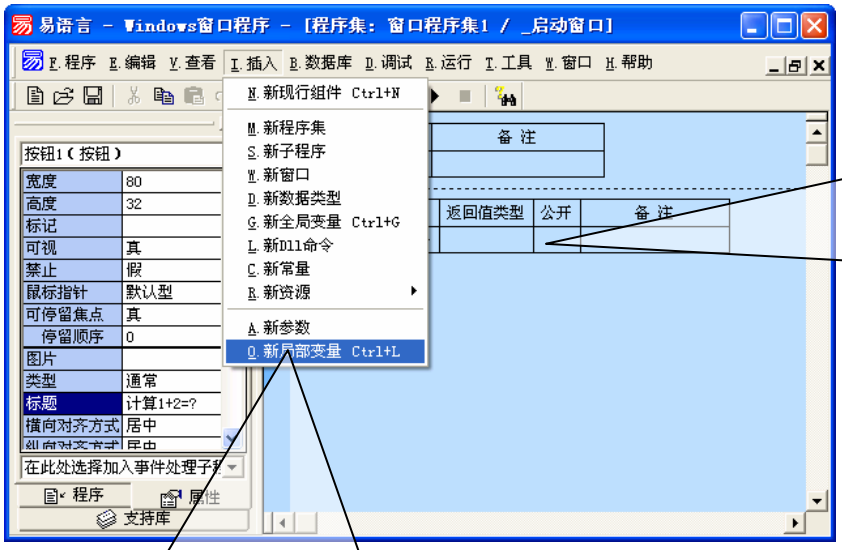
③点击“按钮”组件。

④在窗体上拉出一个按钮组件。形成“按钮1”。



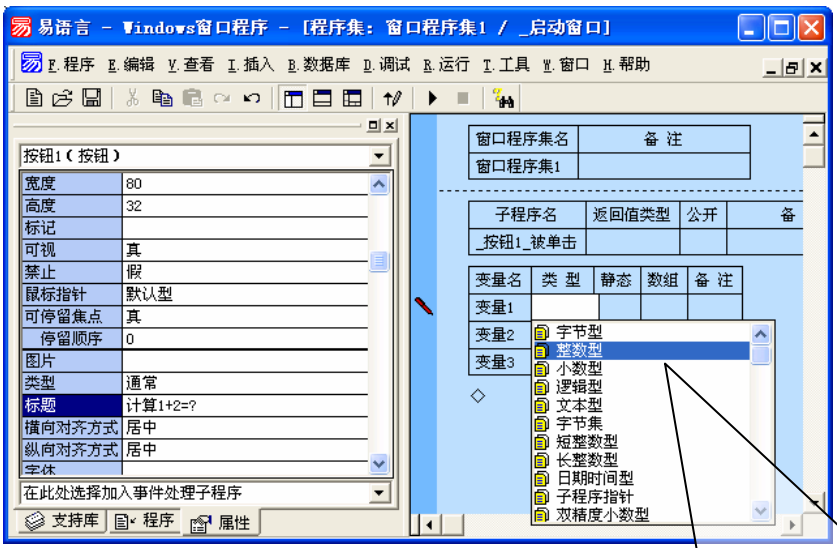
⑤在属性面板中找到标题属性，改内容为“计算1+2=？”。

⑥大家可以看到按钮控件的标题相应的更改了，大家再双击按钮组件，进入程序设计界面。



大家可以看到已自动生成一个名称为“_按钮_1_被单击”的子程序。

⑦用鼠标点击菜单“插入”→“新局部变量”，可以添加变量表。使用组合键[Alt+L]也可以进行添加操作。



⑧在变量表中填写变量名，如为“变量 1”，在类型一栏中使用键盘上的空白键，会弹出一个数据类型的下拉条，从中选择“整数型”。请大家添三个这样的变量。

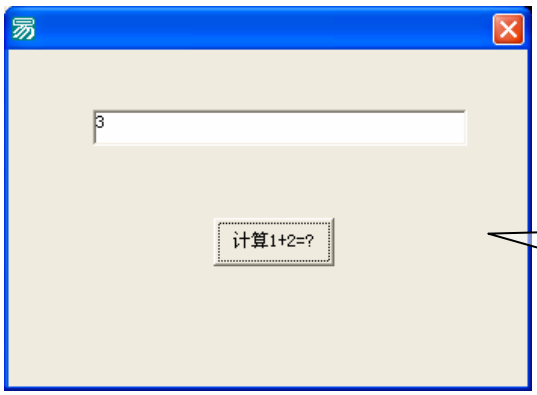
子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	整数型			
变量2	整数型			
变量3	整数型			

```
变量1 = 1
变量2 = 2
变量3 = 变量1 + 变量2
编辑框1.内容 = 到文本 (变量3)
```

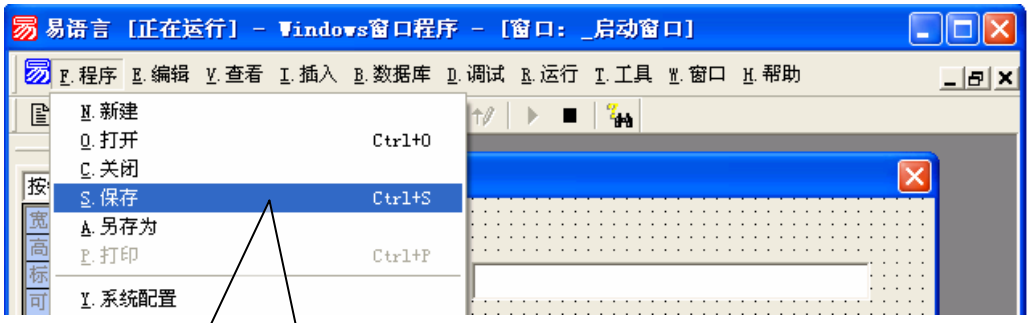
⑨接下来输入四行程序代码：

```
变量 1 = 1

变量 2 = 2
```

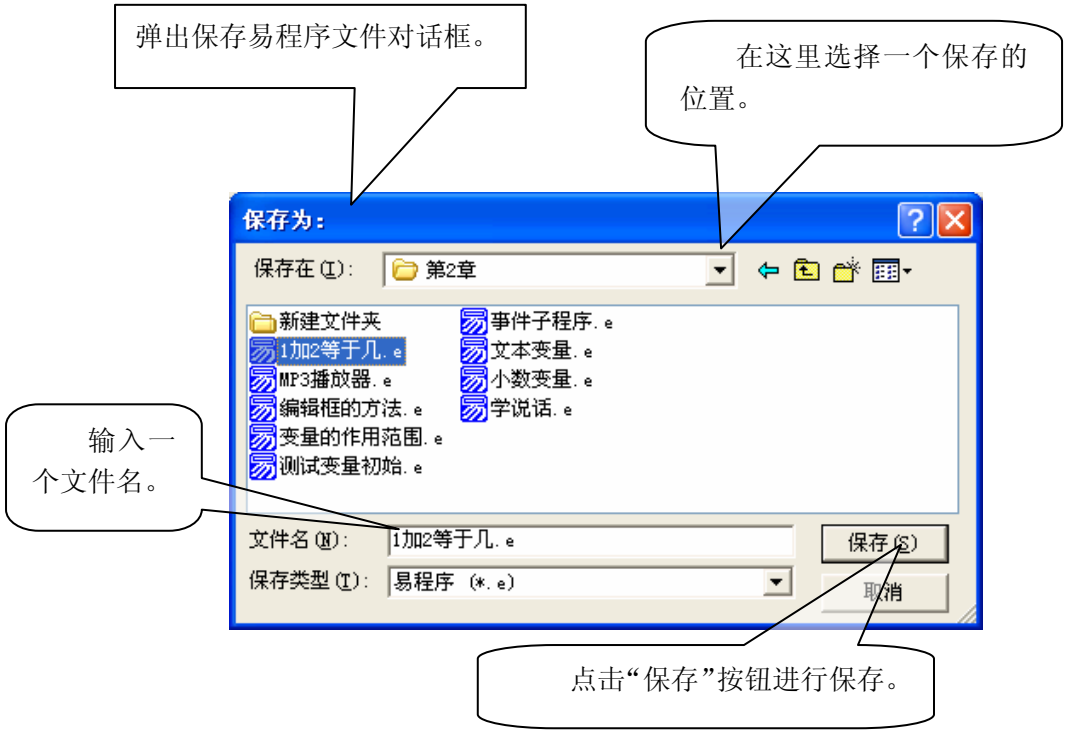


⑩最后就可以使用功能键“F5”键，试运行一下了。

运行后点击按钮，在编辑框中就显示答案了。



最后使用菜单命令保存这个易语言程序。



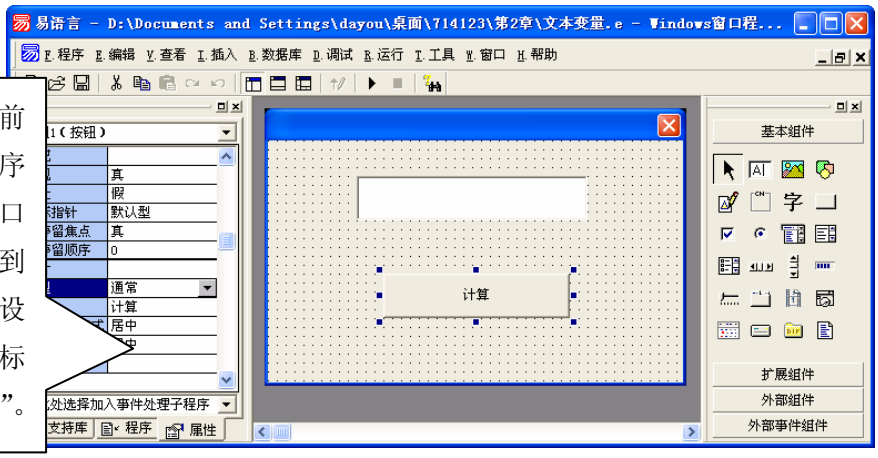
2.2 什么是变量

通过前面一节的学习，大家已建立了第二个易语言程序，这个程序会计算 1+2 等于几。那么在这里会涉及四个小的概念：“变量”、“数据类型”、“事件”及“命令”。下面分别进行解释。

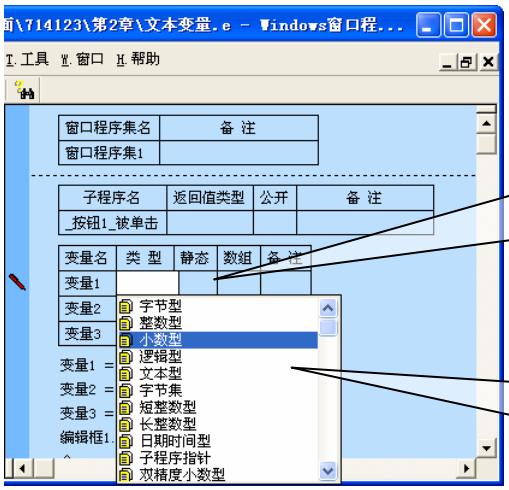
易语言的变量在易语言 3.5 版本之前都叫作容器。

大家可以理解为上街买菜时装菜用的菜篮子。而数据是装在篮子中的各种蔬菜，有青菜、萝卜、黄瓜等等。而各种蔬菜即是不同的数据类型。

了解变量十分重要，下面用一个小例程来说明。

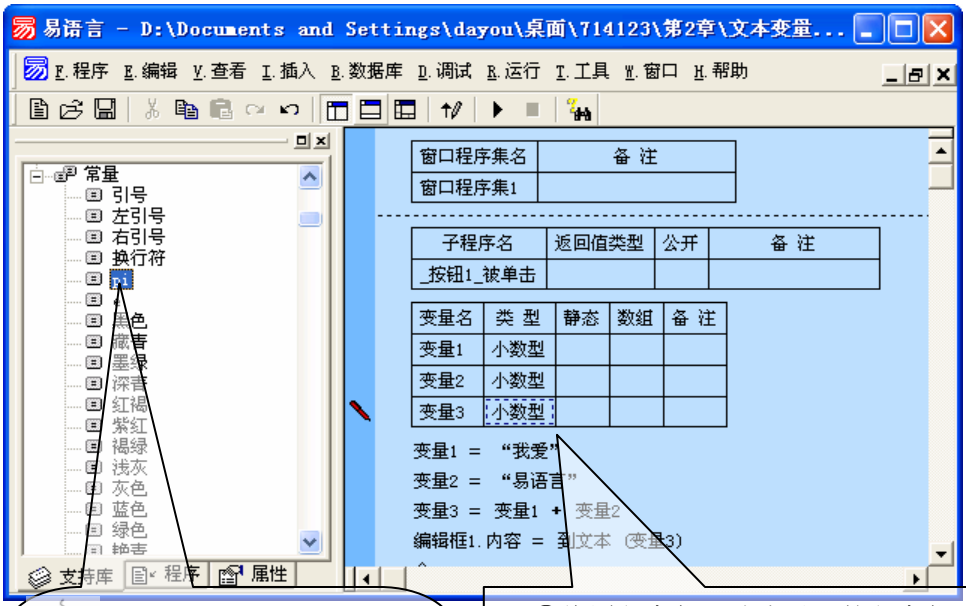


①打开前一节的程序后，使用窗口菜单切换到启动窗口设计界面，改标题为“计算”。



②双击按钮后进入程序代码设计界面。在变量表的类型中按键盘上的空格键，将三个变量的类型改为“小数型”。

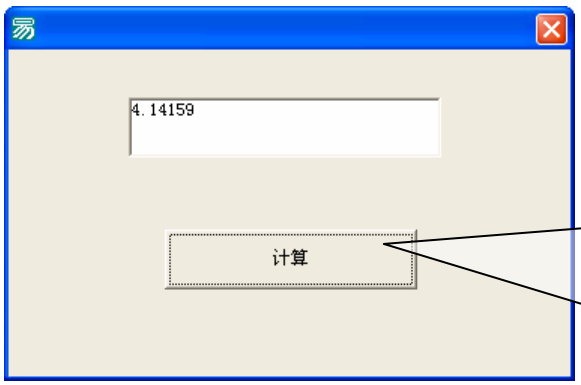
注意：大家可以可以试试看，如果不进行这一步操作会有什么结果。



注意：这里的 pi 是一个常量，这可以在支持库面板的常量表中找到。并且使用常时，前面要加一个“#”号。pi 代表圆周率的 3.14159。

③将原程序代码改为以下的程序代码：

```
变量 1 = 1

变量 2 = #pi
```



④按下 F5 热键，试运行一下。

可以看到编辑框中的结果是：4.14159。

这是变量 1 与变量 2 相加的结果。

最后不要忘记结束程序的运行。

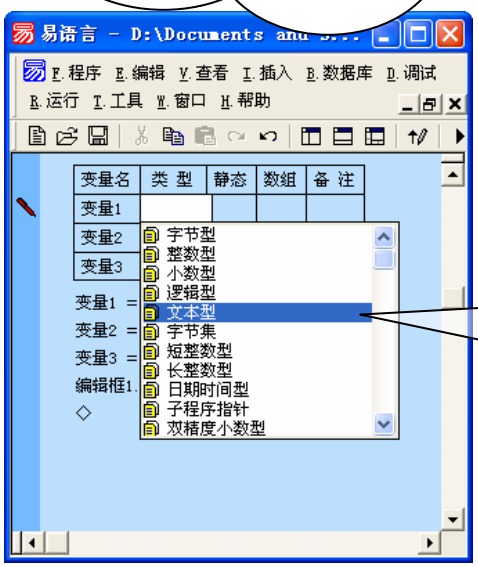
变量与常量的关系：

变量是可以随时进行变化的，也可以由用户改变。

而常量是固定不变的，即有系统固定好的常量，也可以由程序设计者在程序设计时指定常的值。

2.3 变量的数据类型

前面使用过两种数据类型，即整数型与小数型。下面我们通过一个例子，再教大家一种“文本型”的数据类型。通过这个例子，告诉大家还可以有其它的一些数据类型。以及数据类型的初始值等。



打开上节编写的程序，双击按钮组件后，进入程序设计界面。

将三个变量的数据类型都改为文本型。

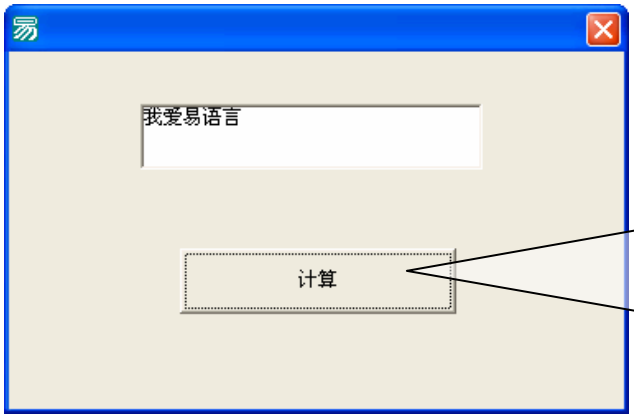
子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1				
变量2	文本型			
变量3	文本型			

```
变量1 = “我爱”
变量2 = “易语言”
变量3 = 变量1 + 变量2
编辑框1.内容 = 到文本 (变量3)
```

将程序内容改为以下程序代码：

```
变量 1 = “我爱”

变量 2 = “易语言”
```



按键盘上的 F5 功能键，可以试运行一下这个程序。

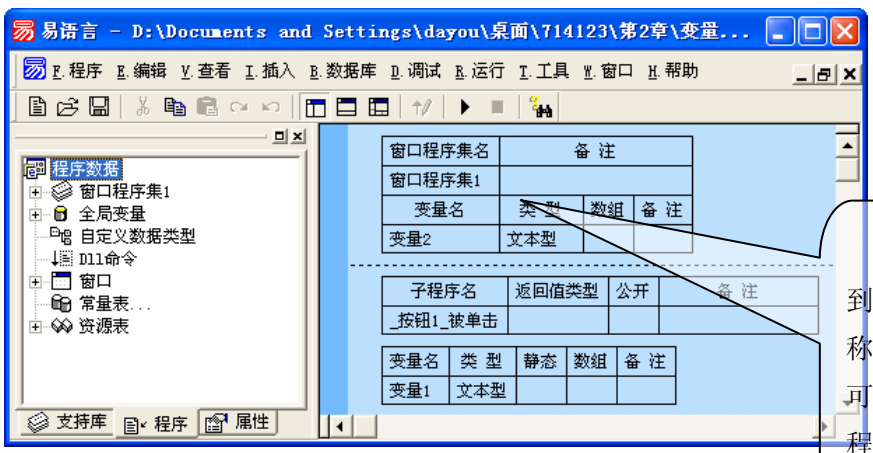
运行后用鼠标点击在按钮，就会在编辑框中显示计算结果“我爱易语言”。

下面列出的是变量与常量的“数据类型”与“数据类型长度”。

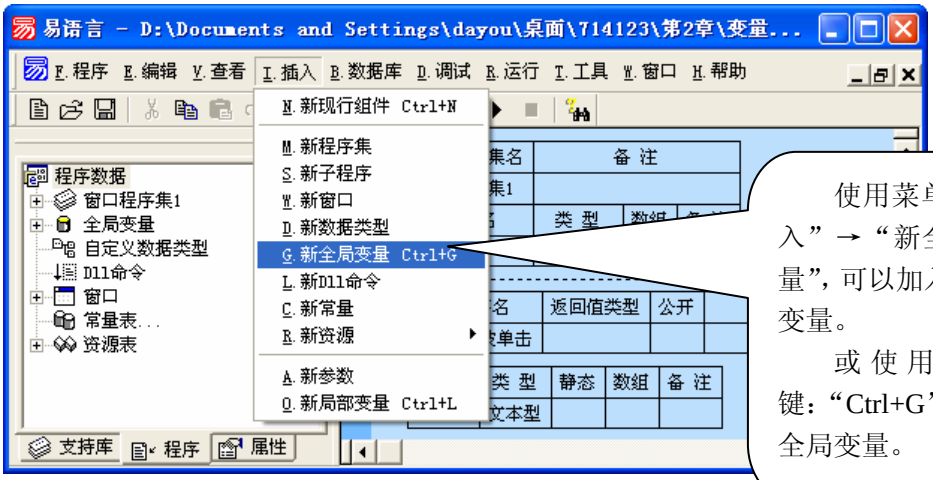
数据名称	数据类型长度
字节型	0~255 个字节
短整数型	-32,768 到 32,767 之间的数值，尺寸为 2 个字节
整数型	-2,147,483,648 到 2,147,483,647 之间的数值，尺寸为 4 个字节
长整数型	-9,223,372,036,854,775,808 到 9,223,372,036,854,775,807 之间的数值，尺寸为 8 个字节
小数型	3.4E +/- 38 (7 位小数) 之间的数值，尺寸为 4 个字节
双精度小数型	1.7E +/- 308 (15 位小数) 之间的数值，尺寸为 8 个字节
逻辑型	“真”或“假”，尺寸为 2 个字节
日期时间型	记录日期及时间，尺寸为 8 个字节
文本型	可记录一段文本，文本由以 0 结束的一系列字符组成
字节集	用作记录一段字节型数据。字节集与字节数组之间可以互相转换，在程序中允许使用字节数组的地方也可以使用字节集，或者相反。字节数组的使用方法，譬如用中括号对（“[ ]”）加索引数值引用字节成员，使用数组型数值数据进行赋值等等，都可以被字节集所使用。两者之间惟一的不同是字节集可以变长，因此可把字节集看作可变长的字节数组
子程序指针	用作指向一个子程序，尺寸为 4 个字节。具有此数据类型的容器可以用来间接调用子程序 字节型、短整数型、整数型、长整数型、小数型、双精度小数型统称为数值型，它们之间的区别在于所容纳数值范围的不同和数据尺寸的不同。

2.4 变量的有效范围

变量分为子程序变量、程序集变量与全局变量三种。它们各有自己的作用范围。为什么要分配不同的作用范围呢？这是因为一方面只在子程序中有效的变量写在子程序中，以利于观察，另一方面，操作系统会在子程序运行后收回内存空间，以节约内存。再说：如果大量的使用全局变量，会占用大量的内存，而且也比较乱，因为有的变量只用到一、两次就不再用了，这样会非常浪费。



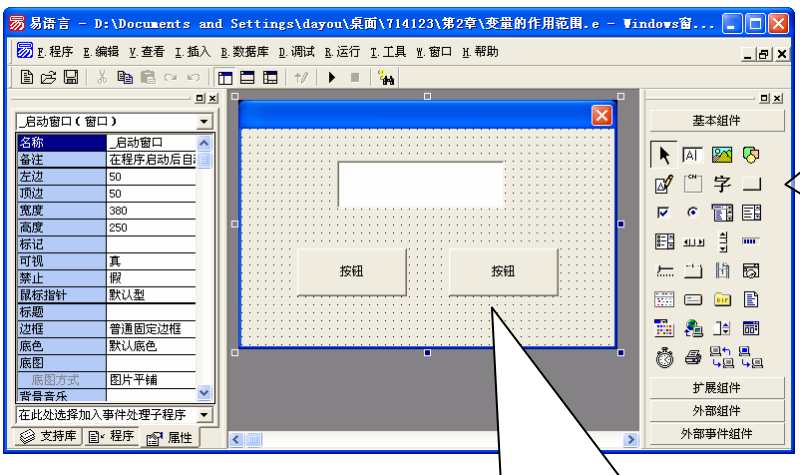
光标移到程序集名称上回车即可添加一个程序集变量。



使用菜单“插入”→“新全局变量”，可以加入全局变量。

或使用快捷键：“Ctrl+G”增加全局变量。

下面还是通过一个例程测试来了解一下变量的作用范围。



①使用上述已建好全局变量、程序集变量、子程序变量的例程。再用“窗口”菜单回到程序设计界面。调整原来的一个按钮，再增加一个按钮。

②双击按钮 1 控件，以进入“_按钮 1_被单击”事件子程序设计界面。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	文本型			

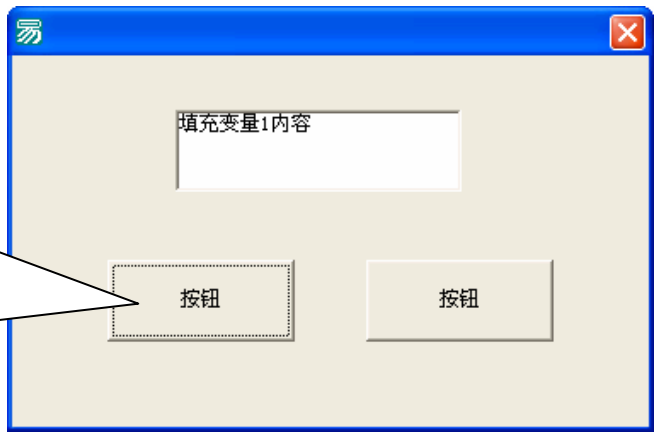
```
变量1 = “填充变量1内容”
编辑框1.内容 = 变量1
```

③在“_按钮 1_被单击”子程序中输入以下程序代码：

```
变量 1 = “填充变量 1 内容”
编辑框 1.内容 = 变量 1
```

按热键 F5，试运行这个程序，查看一下效果。

点击按钮 1 后，可以看到变量 1 的内容在编辑框中显示出来了。

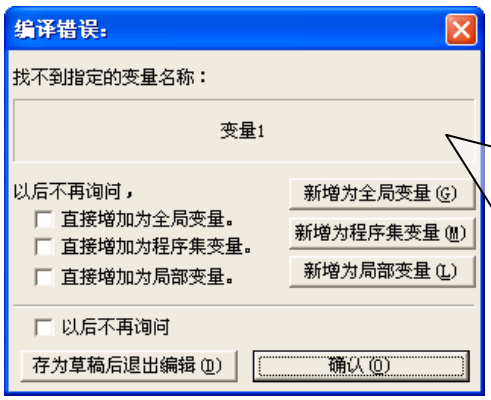


子程序名	返回值类型	公开	备 注
_按钮2_被单击			

```
编辑框1.内容 = 变量1
```

④结束程序的试运行。回到设计界面。双击按钮 2，进入程序设计界面。在“_按钮 2_被单击”子程序中输入以下程序代码：

```
编辑框 1.内容 = 变量 1
```



⑤这时如果回车确认输入，会弹出一个找不到指定的变量名称的错误对话框，这是因为变量 1 只作用于按钮 1，而不能作用于按钮 2。

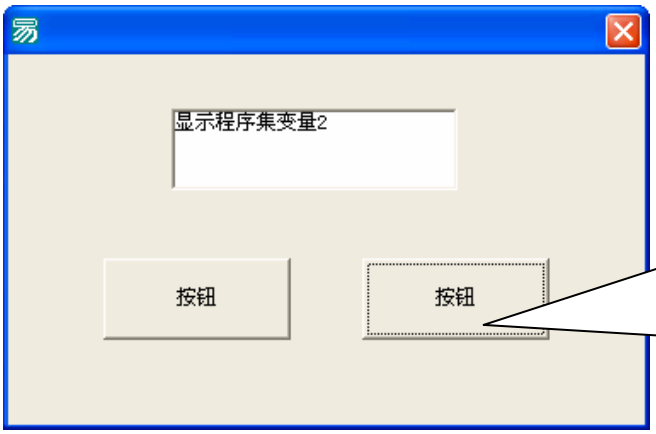
可以改为以下程序代码：

```
变量 2=“显示程序集变量 2”
```

在这里，我们可以看到，由于在窗口程序集中定义过变量2了，所以在当前整个窗口程序集中都可以直接使用。

子程序名	返回值类型	公开	备 注
_按钮2_被单击			

变量2 = “显示程序集变量2”
编辑框1.内容 = 变量2



按热键 F5，试运行这个程序，查看一下效果。

点击按钮 2 后，可以看到变量 2 的内容在编辑框中显示出来了。

对于变量 3 来说，由于是全局变量，因此在程序的任何位置都可以使用。包括不同的程序集与子程序。

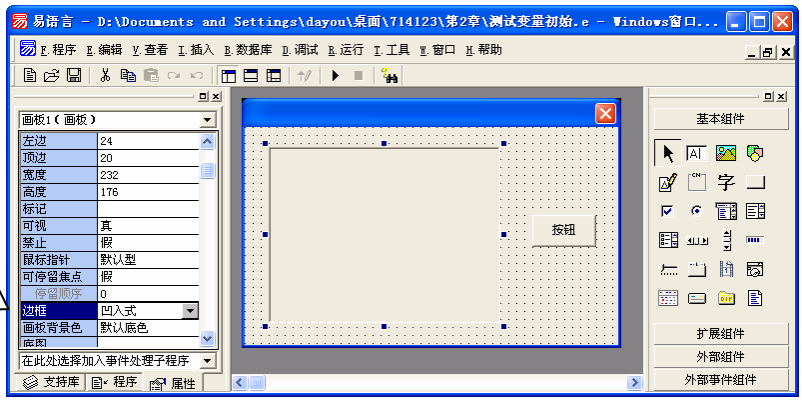
如果您增加了新的窗口，就会为每一个窗口自动生成一个窗口程序集。窗口程序集变量可以作用于窗口内的所有程序，但不能作用于其它窗口程序集的子程序。

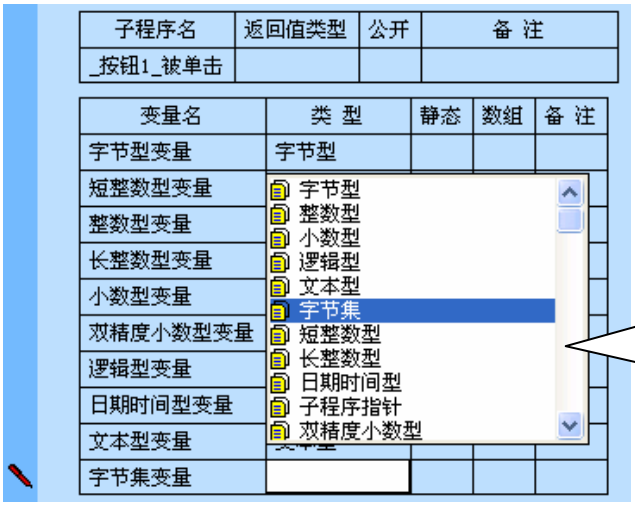
2.5 变量的初始值

如果容器内从来没有被写入过数据，那么此时容器中的内容是什么呢？

下面通过自编写一个小程序，就可以测试出来了。

新建一个易程序，在窗体上放一个画板控件和一个按钮控件。将画板控件的**边框**属性改为凹入式。





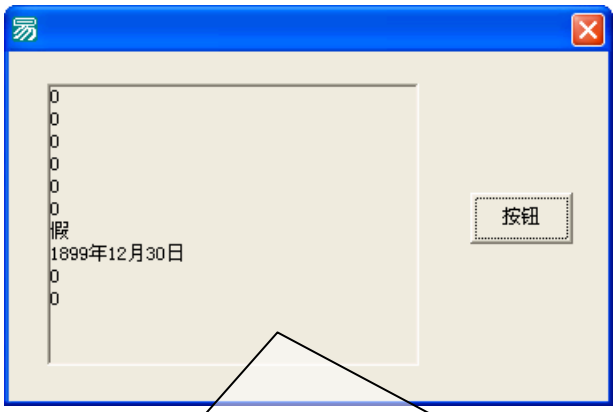
双击按钮控件，进入程序录入界面，顺序加入根据类型名命名的变量，共计 10 个。

如整数类型的变量，变量名即为“整数型变量”，其它类似。

在下面继续输入以下程序代码：

```
画板 1.滚动写行（字节型变量，短整数型变量，整数型变量，长整数型变量，小数型变量，双精度小数型变量，逻辑型变量，日期时间型变量，取文本长度（文本型变量），取字节集长度（字节集变量））
```

```
画板1.滚动写行（字节型变量，短整数型变量，整数型变量，长整数型变量，小数型变量
    ，双精度小数型变量，逻辑型变量，日期时间型变量，取文本长度（文本型变量），取
    字节集长度（字节集变量））
```



按 F5 快捷键试运行，点击按钮，即可得到测试结果。

通过测试，可以知道他们的初始值了，分别说明如下：

- 0 → 字节容器的内容
- 0 → 短整数容器的内容
- 0 → 整数容器的内容
- 0 → 长整数容器的内容
- 0 → 小数容器的内容
- 0 → 双精度小数容器的内容
- 假 → 逻辑容器的内容
- 1899 年 12 月 30 日 → 日期时间容器的内容
- 0 → 文本容器中文本的长度
- 0 → 字节集容器中字节的数目

由上面的显示结果可以知道：所有数值型容器的初始值都为 0 ，逻辑型容器的初始值为假，日期时间型容器的初始值为 1899 年 12 月 30 日，文本型容器的初始值为长度为 0 的空文本，字节集容器的初始值为空字节集。如果容器为数组，其每个数组成员的初始值都与单个容器相同。

2.5 编写一个 MP3 播放器

在本节，您可以跟着步骤制作一个简单的 MP3 播放器。

做这个程序，大约需如下几步：1、启动易语言，新建一个易程序； 2、设计程序界面； 3、写代码；4、运行编好的程序；5、生成可执行文件。这也是编写一个易语言程序的通用步骤，其中第 2、3 步可能重复多次，以修改与加强程序功能。

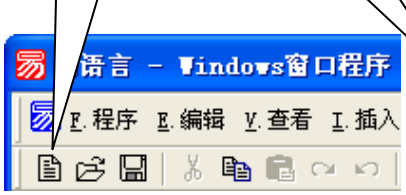
第一步：新建一个易程序。

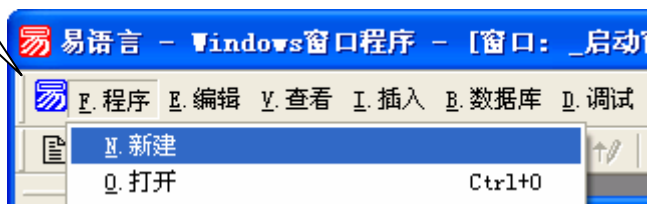
实际上有三种方法可以新建易程序：1、在没有运行易语言时，双击易语言图标，即可启动弹出新建对话框，并在对话框中选中“Windows 窗口”图标，点击确定即可。2、使用菜单“程序”→“新建”。3、使用快捷按钮。均可弹出新建对话框。

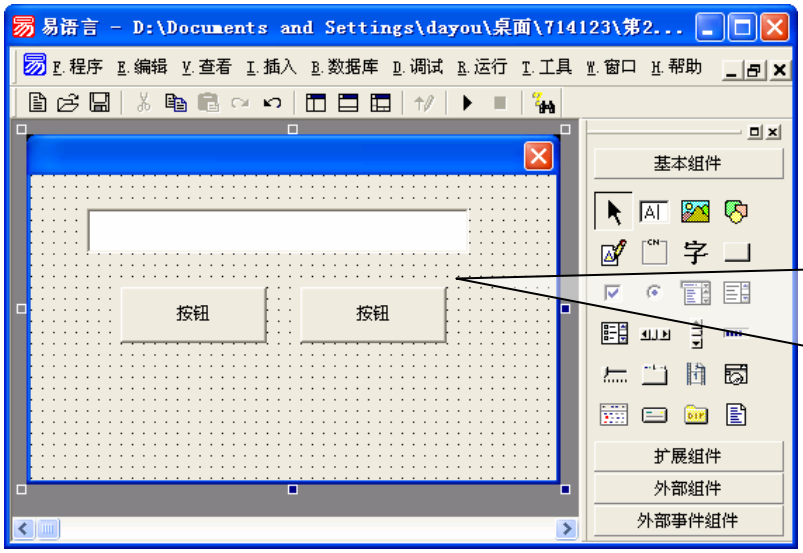
以后新建易程序均照此三种操作。

快捷按钮

使用菜单





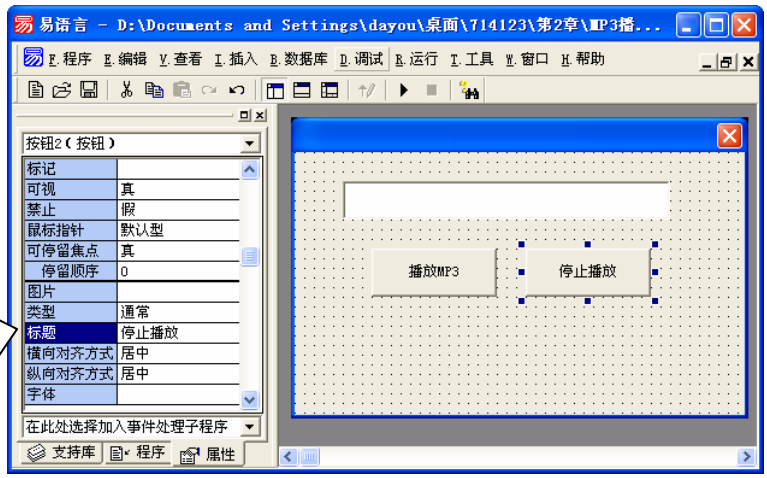


第二步：设计程序界面。

分别在新窗体中画一个编辑框和两个按钮控件。

选中按钮后，打开属性面板。

分别将这两个按钮的**标题**属性改为“播放 MP3”和“停止播放”。



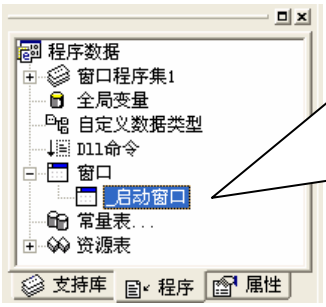
子程序名	返回值类型	公开	备 注
_按钮1_被单击			

播放MP3 (1, 编辑框1.内容)

第三步：写代码

双击标题为“播放 MP3”的按钮，进入代码编辑区后输入以下代码：

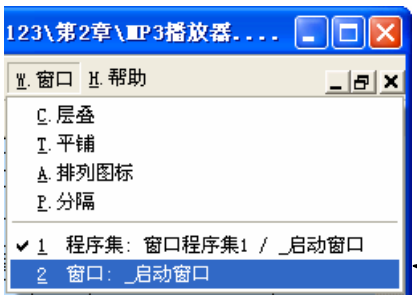
```
播放MP3（1，编辑框1．内容）
```



下面一步就要双击标题为“停止播放”的按钮，为它写代码了。可问题是，我们正处于**代码编辑区**中，根本看不到**窗体设计区**，更不要说双击其中的按钮了！所以，目前的当务之急就是，切换到**窗体设计区**。切换的方法有三：

1、利用**工作夹**

首先将**工作夹**中的**程序面板**切换到前台，然后单击“窗口”前的“+”号使其变为“-”，这时会发现“窗口”下面又出现了一个分枝：“_启动窗口”，用鼠标双击它，就可以将操作环境从**代码编辑区**切换到**窗体设计区**。



2、利用“窗口”菜单

易语言主菜单中的“窗口”菜单如右图所示，选择“窗口：_启动窗口”即可切换到**窗体设计区**。

3、利用热键 Ctrl+Tab 也可以在**代码编辑区**和**窗体设计区**之间切换。

以后切换均照此三种方法之一操作。以后不再讲述。

子程序名	返回值类型	公开	备 注
_按钮2_被单击			

停止播放 ()

双击标题为“停止播放”的按钮，自动切换到“_按钮 2_被单击”子程序，在光标所在行输入：

停止播放 ( )

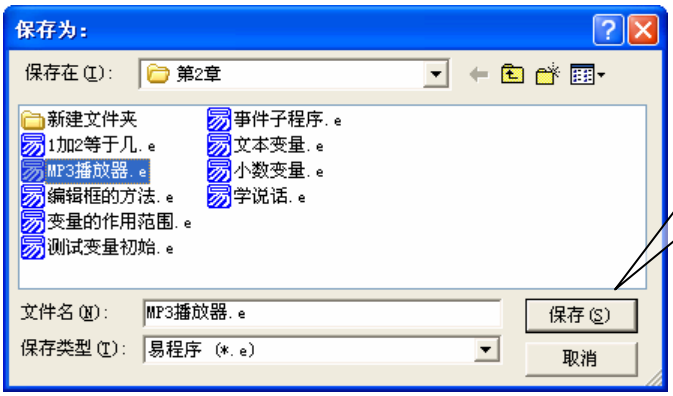
第 4 步：运行编好的程序

有三种方法可以实现运行例程：1、选择主菜单“运行”→“运行”。

2、或单击工具栏上的“运行”按钮。

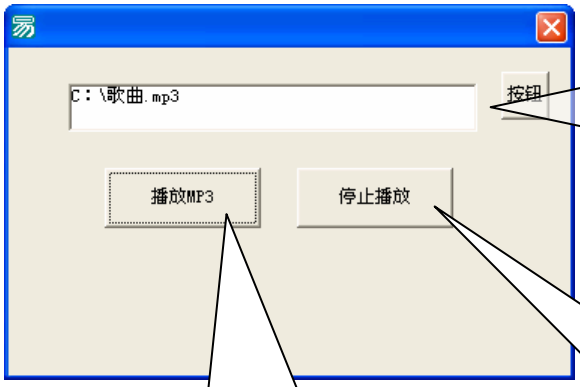
3、或者按热键 F5 都可以试运行当前的程序。

最后使用菜单“程序”→“保存”，保存这个文件。



填入文件名后，点击“保存”按钮即可。

新建易程序、切换界面、保存易程序、运行易程序都作为基本的操作，以后不再重复介绍了。

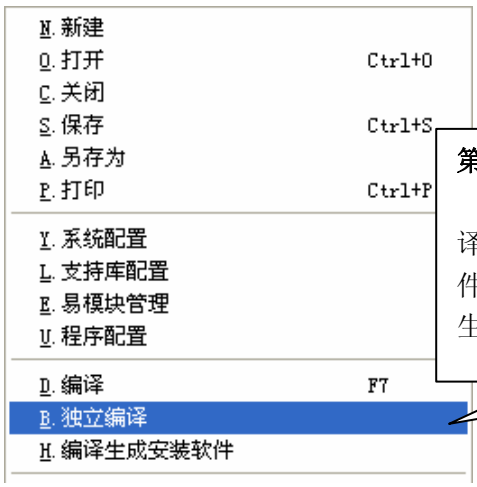


运行后的操作：

先找一首歌，记下路径文件名。按热键 F5 试运行当前的程序。在编辑框中填入歌曲的全路径文件名，例如：c:\歌曲.mp3

点击标题为“播放MP3”的按钮，应该可以听到音乐了吧。（电脑必须配声卡及音箱）

单击按钮“停止播放”后再输入另一个 MP3 文件名，再单击“播放 MP3”，是不是又一首 MP3 响起了。



第 5 步：生成可执行文件

选择菜单“程序”→“编译”或“独立编译”即可将本程序编译为可执行文件（EXE 文件）。建议用“独立编译”编译就可以了，这样生成的可执行文件可以直接拷贝给别人用。

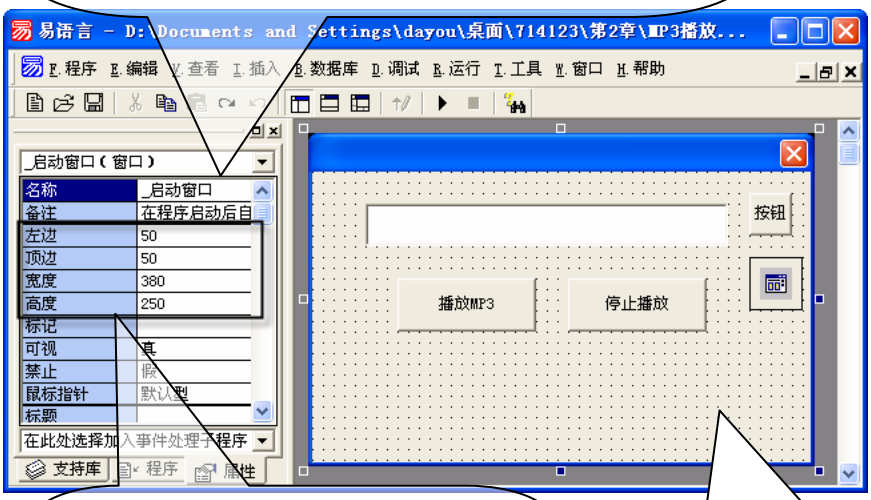
至此，一个简单的 MP3 播放器就好了。大家可以任意在编辑框中填入 MP3 歌曲的全路径，再点击播放就可以听到音乐了。在课后练习中，会教大家更改一个界面，以及使用通用对话框找歌曲的名字。

2.5 认识窗口、按钮、编辑框

在前面的章节中，大量用到了三个常用组件，认真地学习这三个基本的组件，也有利于后面的学习。

本节将认识窗口、按钮、编辑框三个常用组件的属性、事件、与方法。

在新建的易程序中，总有一个“_启动窗口”，在属性面板中最上排有一个名称属性为:“_启动窗口”。名称属性是窗口组件的识别字,一般要取一个有意义的名称。而且“_启动窗口”是首次运行的窗口，如果没有将不能运行 ，所以是不能更改的。



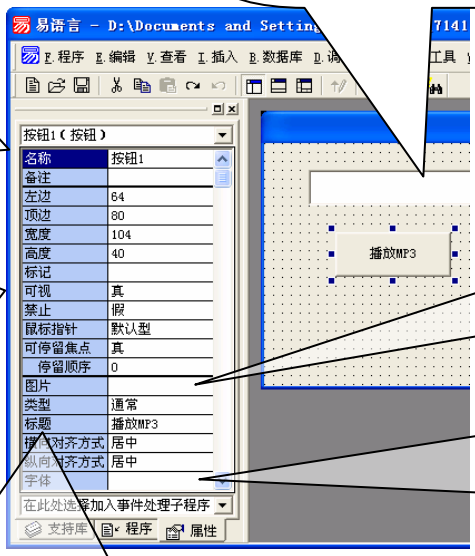
下面有四个属性是表示窗口的坐标的。更改宽度值与高度值，可以改变组件的大小。

大家试试激活窗口中的其他组件，也有这些属性可改变。

每个组件被激活后，就会出现 8 个夹点，直接用鼠标拖动这些夹点，就可以改变组件的尺寸了。

大家激活按钮组件，也可以看到8个夹点，可以直接用鼠标拖动，改变它的尺寸。

按钮组件，也有**名称**属性与**坐标**属性。可为按钮组件重取一个容易理解的名字。

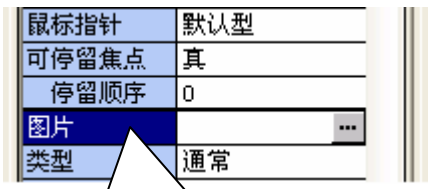


按钮的**可视**属性表示运行时按钮是否可见。**禁止**属性表示运行时是否可操作。大家试分别改一下，试运行看看效果。

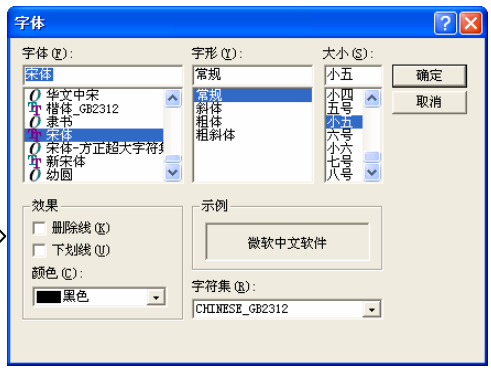
按钮的**图片**属性可为按钮表面更换一张图

按钮的**字体**属性可改变按钮标题文字的大小风格等。

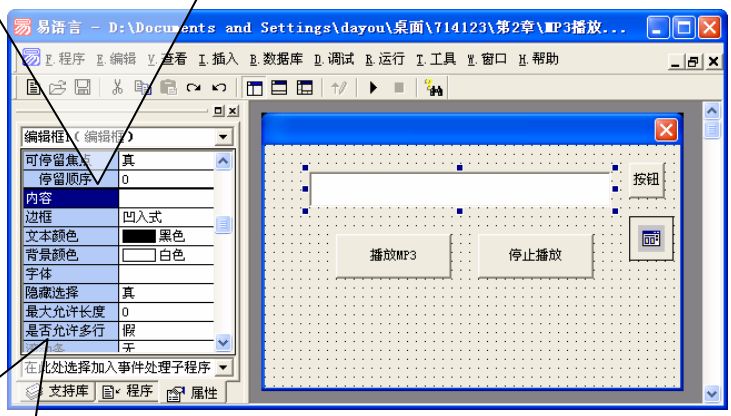
按钮的**标题**属性是显示在按钮上的文字。大家可以试着改一下，再看看按钮上文字的变化。



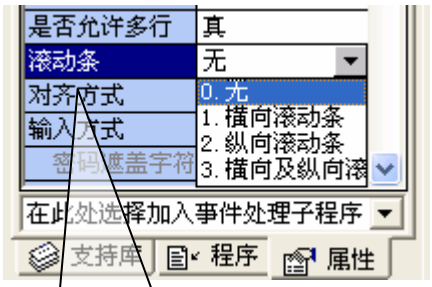
分别激活**图片**属性与**字体**属性后，会出现一个按钮，点击后就会弹出另一个对话框。从中进行选择。



大家再激活编辑框组件，可以看到编辑框组件没有**标题**属性，只有一个**内容**属性。这表示当程序运行时，为**内容**属性的可由用户改变内容，而**标题**属性不可直接修改。大家试运行一下，可以直接在编辑框中填写内容，而按钮与窗口却不行。



是否允许多行属性为假时，所有输入只显示为一行，为真时，可以显示为多行。



在是否允许多行属性为真的情况下，**滚动条**属性可操作，点击后会弹出一个下拉菜单，可选其中的**纵向滚动条**，这样文字过多时，可通过滚动条查看更多的文字。

输入方式属性被改变时，也会弹出一个下拉菜单，大家可以试着分别选择，试运行一下，看看效果。

其中“密码输入”方式运行时显示的是星号，可以应用于口令输入。

上面对窗口、按钮、编辑框三个基本组件的基本属性进行了介绍，其它的属性大家可以自己试着进行改变，再试运行，即可看到效果，也可以激活某一属性后即按下 F1 帮助键，得到与此属性相关的帮助。

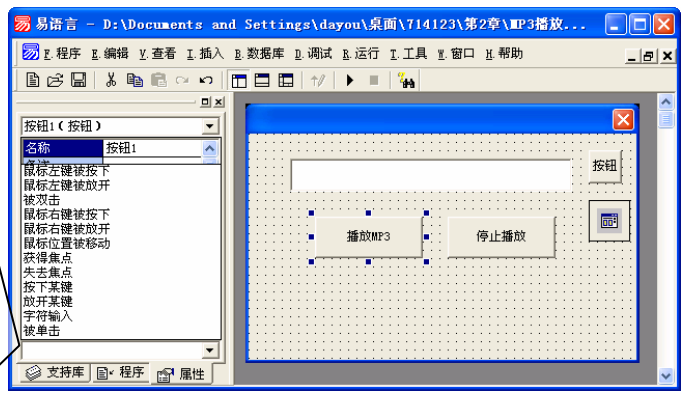
2.5 认识事件子程序

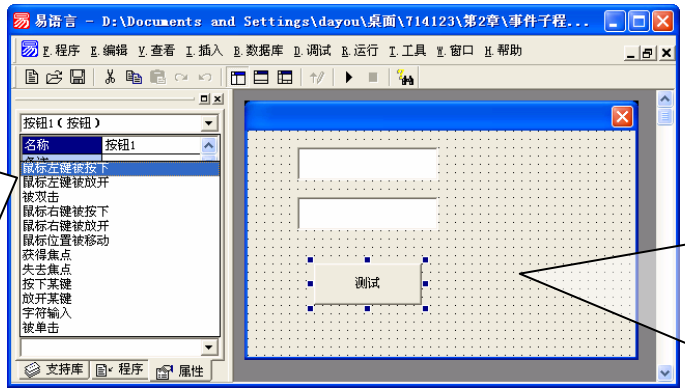
在前面的章节中，双击按钮控件得到的按钮“被单击”子程序实际上就是一个事件子程序。

在本节将系统地了解事件子程序。

激活一个按钮，在属性面板最下方的下拉项中选择“被单击”后，也会自动生成“_按钮 1_被单击”子程序。这就是一个事件子程序，下拉项中的所有项目都是这个按钮的事件。

按钮不仅可以接受鼠标左键单击，还可以接受鼠标右键单击，以及双击等，都可以通过这个选项生成事件子程序。





①新建一个易程序，放置两个编辑框，与一个按钮控件，改按钮控件的标题属性为“测试”。

②分别选择属性面板中的四个事件，以自动生成事件子程序。

子程序名	返回值类型	公开	备 注		
_按钮1_鼠标左键被按下	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

编辑框1.内容 = “_按钮1_鼠标左键被按下”

子程序名	返回值类型	公开	备 注		
_按钮1_鼠标左键被放开	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

编辑框1.内容 = “_按钮1_鼠标左键被放开”

子程序名	返回值类型	公开	备 注		
_按钮1_鼠标右键被按下	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

编辑框1.内容 = “_按钮1_鼠标右键被按下”

子程序名	返回值类型	公开	备 注		
_按钮1_鼠标右键被放开	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

编辑框1.内容 = “_按钮1_鼠标右键被放开”

③这四个事件分别是：“鼠标左键被按下”、“鼠标左键被放开”、“鼠标右键被按下”、“鼠标右键被放开”。分别形成四个事件子程序。

④分别在这四个事件子程序中输入程序代码，用编辑框1显示一些文字，以演示鼠标产生动作后会发生的事件。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

编辑框2.内容 = “_按钮1_被单击”

子程序名	返回值类型	公开	备 注		
_按钮1_被双击	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

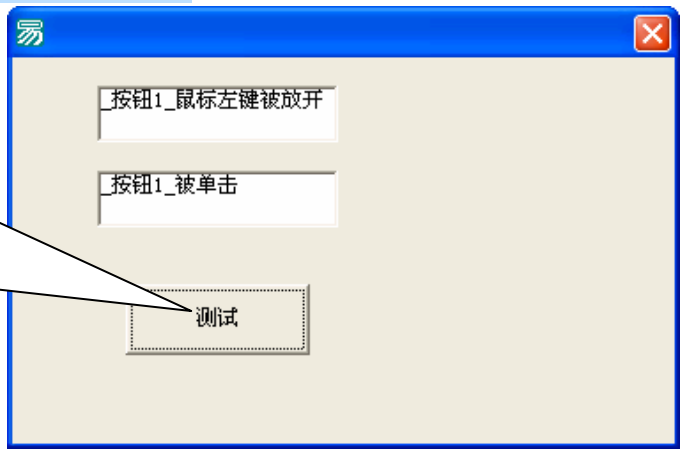
编辑框2.内容 = “_按钮1_被双击”

⑤可再加两个按钮事件子程序：“被单击”及“被双击”事件，生成事件子程序。

程序表示将在编辑框2中显示被单击及被双击事件的结果。

⑥按F5试运行这个测试程序。

可以使用鼠标左键与鼠标右键进行点击的动作，以测试效果。

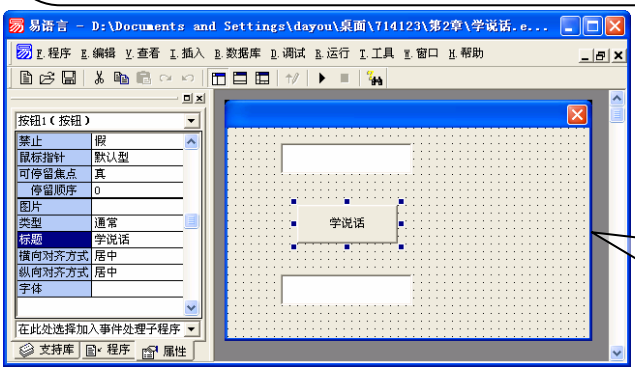


通过以上测试，大家可以发现，无论鼠标按下，与松开，都会产生事件，并且会在编辑框中显示不同的特定文字。鼠标左键与鼠标右键按下后，显示的文字也不同，单击与双击也会有不同的显示。

这些动作即是事件，它们产生的子程序即是事件子程序。

如果大家将事件子程序中的程序代码换为其它的程序代码，那么就会产生其它的运算结果，并可通过编辑框显示出来。

下面跟着作两个小练习，以理解是什么是事件。



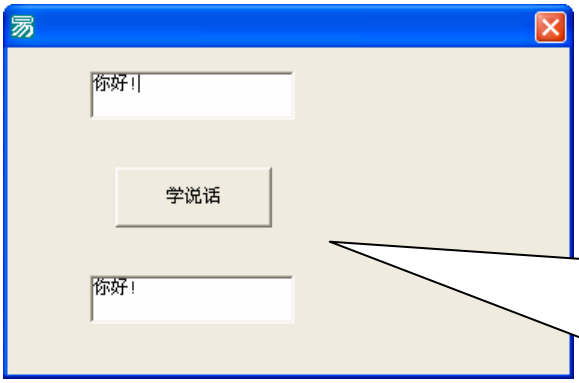
①首先新建一个易程序，在启动窗口中放两个编辑框和一个按钮。并将按钮的标题改为“学说话”。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

编辑框2.内容 = 编辑框1.内容

②在“_按钮 1_被单击”事件子程序中输入以下程序代码：

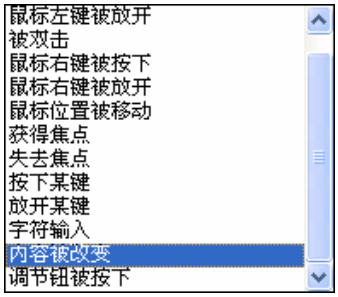
```
编辑框2.内容 = 编辑框1.内容
```

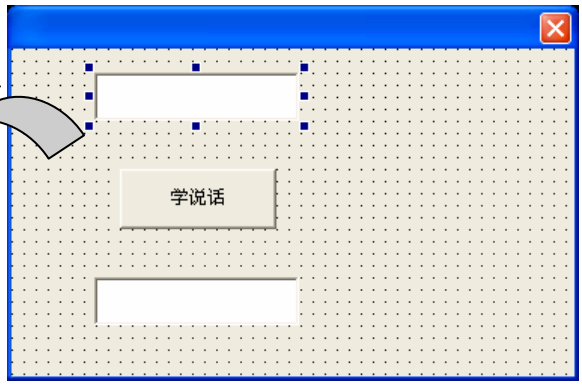


③按 F5 键试运行，在最上面的编辑框中输入文字，再点击按钮，即可以看到另一个编辑框显示了相同的文字。

如果想在一个编辑框中输入，立即显示在另一个编辑框中，怎么办呢？那么跟着步骤再来吧！请结束试运行。请结束程序运行，回到设计界面。

④激活编辑框 1，在属性面板的下拉菜单中选择“内容被改变”事件。松开鼠标后就会自动进入程序设计界面。





子程序名	返回值类型	公开	备 注
_编辑框1_内容被改变			

编辑框2.内容 = 编辑框1.内容

⑤在“_编辑框 1_内容被改变”事件子程序中输入以下程序代码：

```
编辑框2.内容 = 编辑框1.内容
```

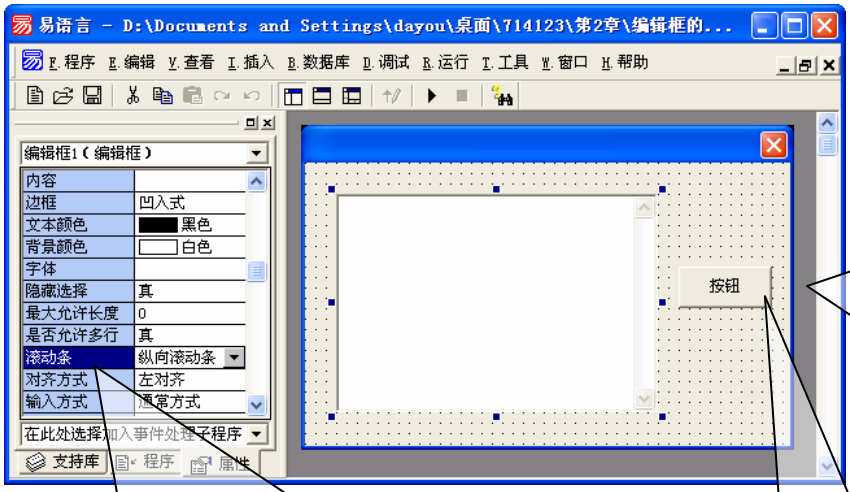
⑥再次试运行，可以在上排的编辑框中输入，立即就会显示在下排编辑框中了。

每个控件都有自己的事件，并通过选择属性面板中的事件下拉列表，可以自动生成事件子程序。通过了解更多的事件子程序与更多的命令，就可以编写复杂一些的程序了。

2.8 认识组件的方法

前介绍了组件的属性、事件，还有一个重要的概念就是组件的方法。

组件的方法也可看作是针对于组件的命令，只不过命令一般都会非常通用地作用于程序的全部环节，而方法只针对于特定的组件。



①新建一个易程序，添加一个编辑框与一个按钮组件。

②将编辑框的“是否允许多行”属性改为“真”。“滚动条”属性改为“纵向滚动条”。

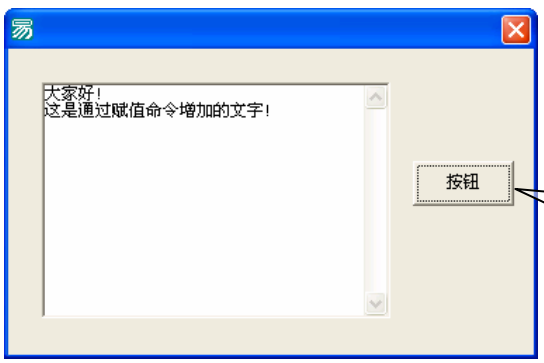
③双击按钮控件，进入被单击事件子程序。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

```
编辑框1.加入文本 (“大家好!” + #换行符)
编辑框1.加入文本 (“这是由编辑框的加入文本方法增加的文字!” + #换行符)
```

④输入以下程序代码。

```
编辑框 1.加入文本 (“大家好!” + #换行符)
```



⑤按 F5 快捷键运行后，点击按钮，可以看到，程序中显示了两行文字。

虽然可以用以下的方法实现：

```
编辑框 1.内容 = “大家好!” + #换行符

编辑框 1.内容 = 编辑框 1.内容 + 这是通过赋值命令增加的文字!” + #换行符
```

但两者是有区别的，前面是使用了编辑框独有的方法显示文字。只有编辑框有这种方法，后面是使用了赋值语句，而赋值语句在所有的程序中都适用。请大家一定要注意这两者的区别。

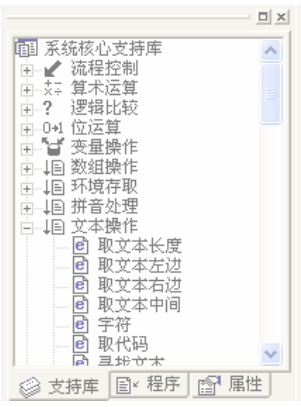
```
编辑框1.内容 = “大家好!” + #换行符
编辑框1.内容 = 编辑框1.内容 + “这是通过赋值命令增加的文字!” + #换行符
```

2.9 课后练习

（1）多练习在窗口设计与程序界面设计之间的切换方法，可以通过易语言主菜单的窗口菜单切换，也可以通过程序面板进行切换。

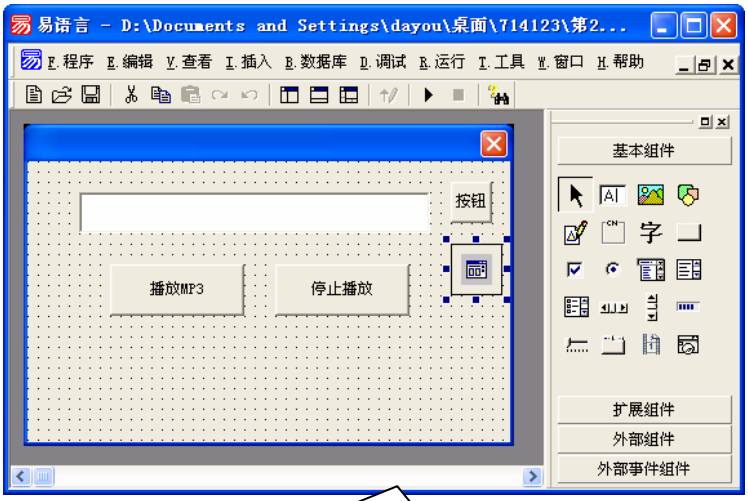
（2）将书中的例子程序“1+2=？”、“MP3 播放器”、“学说话”例程，自己试着重新制作一遍。如不能独立完成，请教你的同学后完成。最后编译为 EXE 可执行文件互相测试。

（3）观察支持库面板中的命令分类，展开后选择一条命令后，再按 F1 帮助键，看看帮助面板中的说明，简单了解这些命令的功能。



（4）为全程 MP3 播放器中增加通用对话框组件，以实现打开任意的 MP3 文件。

打开书中的简单例程，再添加一个按钮，再加入一个通用对话框。



双击新增加的按钮，输入程序代码。最后运行即可。

子程序名	返回值类型	公开	备 注
_按钮3_被单击			

```
如果真 (通用对话框1.打开 ())
    编辑框1.内容 = 通用对话框1.文件名
```

第 3 章 “易语言”的命令

本章主要介绍“易语言”的命令概念，并举出一个大小数判断的例子，介绍判断语句，以及介绍选择语句和循环语句。

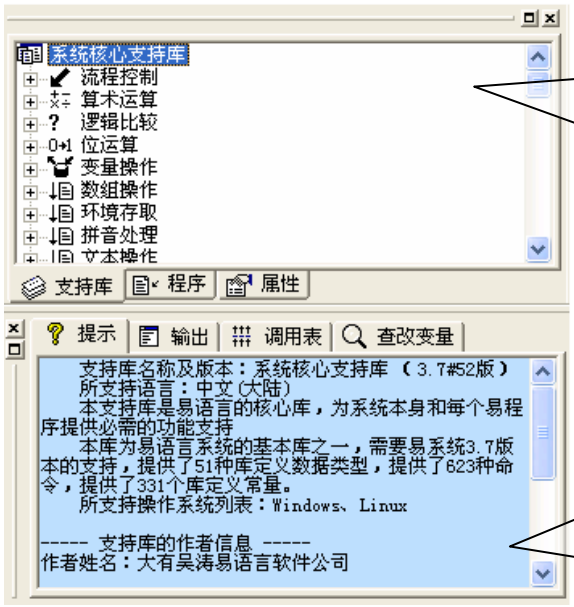
本章学习内容：

命令是比较重要的概念。**程序实际上是由很多的命令组成的。**

“易语言”依靠众多的命令支持着程序的运行。若干命令即组成程序。

“易语言”提供了五百种以上的命令供用户随时调用。通过本章的学习，可以熟练使用命令以及查看命令的即时帮助文件。学习查看命令的即时帮助是本书的教学目的之一，因为“易语言”还在不断的增加命令，对于新增加的命令，在你拿到最新版本时，就可以通过这样的方法去学习命令的使用方法。

3.1 初识命令



打开支持库面板，用鼠标点击系统核心支持库，再按下 F1 热键。可以在提示面板中得到相关的支持库信息。

通过上述的操作后，就可以看到您当前的版本提供的所有命令条数了。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	整数型			
变量2	整数型			
变量3	整数型			

```
变量1 = 1
变量2 = 2
变量3 = 变量1 + 变量2
编辑框1.内容 = 到文本 (变量3)
```

如前面章节中“1+2 等于几”例程中所提到的：

编辑框 1.内容 = 到文本 (变量 3)

其中“到文本（）”就是一个命令，这个命令可以接收一些数据以供其处理，这些数据被称为参数。这个命令就接收了一个数值参数“变量 3”，并将这个数值转换为文本型，交由编辑框显示。

命令是由系统提供的能够完成某一特定功能的指令。它在“易语言”中的书写格式为：

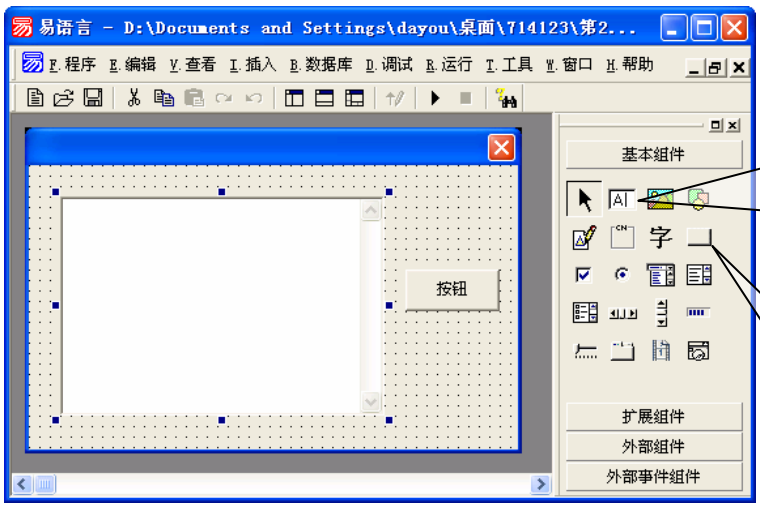
```
命令名称 （参数，.....）
```

一组命令就形成了程序。

命令名称是程序中调用时所使用的名称。

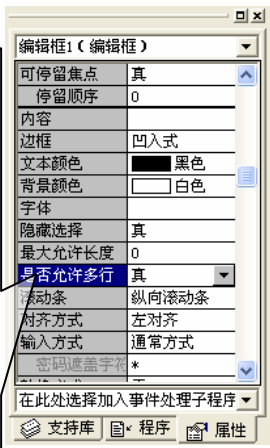
命令所能接收参数的数目和各参数的数据类型由命令本身所决定。所有参数必须用括号一起括住，多个参数之间用逗号隔开。命令执行完毕后还可能返回数据，是否返回数据及所返回数据的数据类型同样由命令本身所决定。如 “到文本（）”命令就将返回文本型数据。

下面通过具体编程来理解上述概念。这个程序列举了几个简单命令的用法。



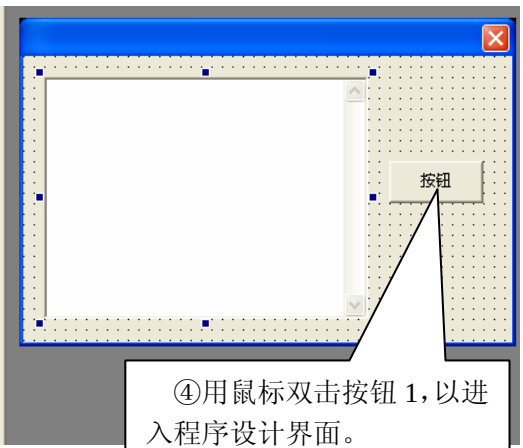
①新建一个易程序，选择编辑框组件，在窗口中拖拉出一个编辑框 1 组件。

②选择按钮组件，在窗口中拖拉出一个按钮 1 组件。



③激活编辑框控件，再展开属性面板，从中找到“是否允许多行”属性，将之改为“真”。

将“滚动条”属性改为“纵向滚动条”。



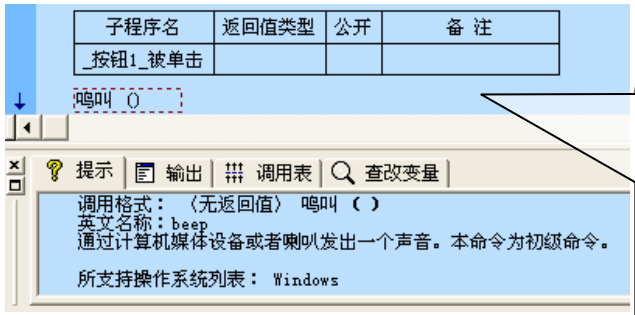
④用鼠标双击按钮 1，以进入程序设计界面。

⑤依次输入以下的程序代码：

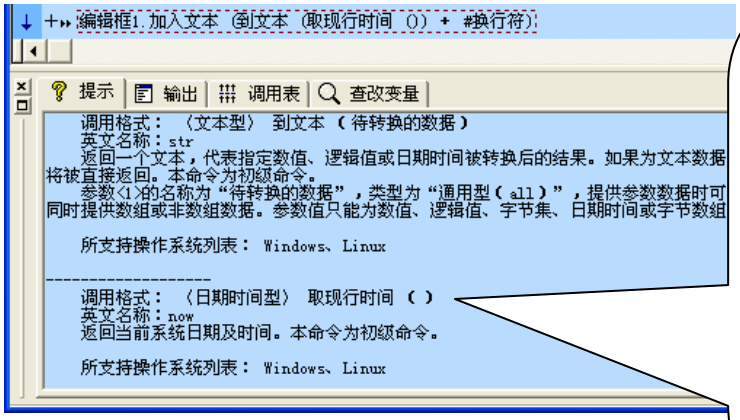
```
鸣叫 ()

编辑框 1.加入文本 (到文本 (取现行时间 ()) + #换行符)

编辑框 1.加入文本 (数值到金额 (100, 假) + #换行符)
```

初学者可能对第二、三行语句的首部不大理解，稍后会讲到，现在只需要知道此语句就是将取现行时间、数值到人民币这两个命令的返回数据显示出来就可以了。

在上面的程序中使用了 3 个不同的命令：“鸣叫（）”、“到文本”、“取现行时间（）”、“数值到人民币（）”。

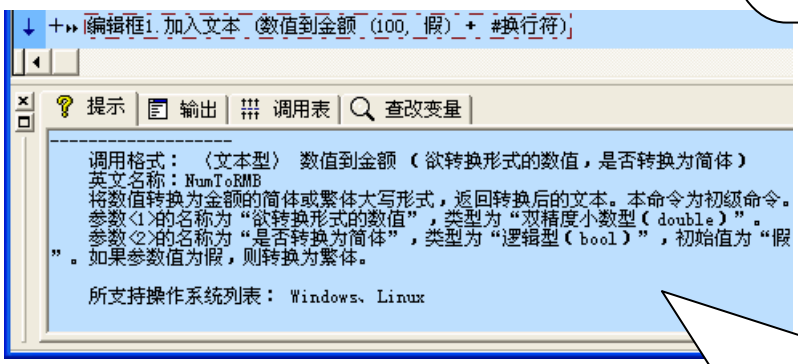


对于“鸣叫”命令，通过将命令行激活，按 F1 热键查看其解释，可以了解到它既不接收参数也不返回数据，仅用作完成发声功能，所以它的参数部分是空的。

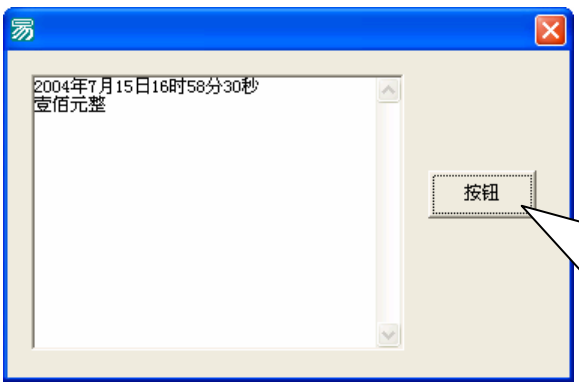


“取现行时间”命令将返回一个日期时间型数据，它也没有参数。

“到文本（）”命令将返回一个文本型数据，它的参数就是“取现行时间”命令返回的一个日期时间型数据。“到文本（）”命令接受了日期时间型数据，并将之转换为文本。



“数值到人民币”命令返回文本型数据并且接收两个参数：参数 1 类型为双精度小数型，名称为“欲转换形式的数值”；参数 2 类型为逻辑型，名称为“是否转换为简体”，其默认值为“假”。由于在程序中没有为参数 2 提供数据，所以系统自动取用其默认值。



⑥按快捷键 F5 键，试运行这个程序，点击按钮，可以看到运行的结果。

查看完成后，请结束程序试运行。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

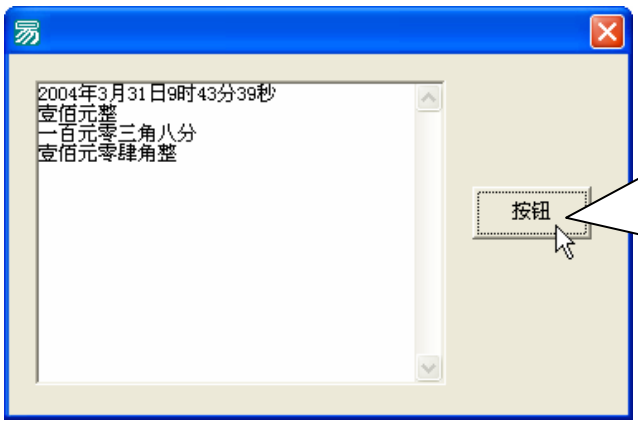
变量名	类 型	静态	数组	备 注
变量1	小数型			

```
鸣叫 ()
编辑框1.加入文本 (到文本 (取现行时间 ()) + #换行符)
编辑框1.加入文本 (数值到金额 (100, 假) + #换行符)
◇
变量1 = 100.38
编辑框1.加入文本 (数值到金额 (变量1, 真) + #换行符)
编辑框1.加入文本 (数值到金额 (四舍五入 (变量1, 1), 假) + #换行符)
```

⑦按[Ctrl+L]键，加入一个名称为“变量 1”的小数型局部变量。

⑧在子程序尾部继续添加以下语句：

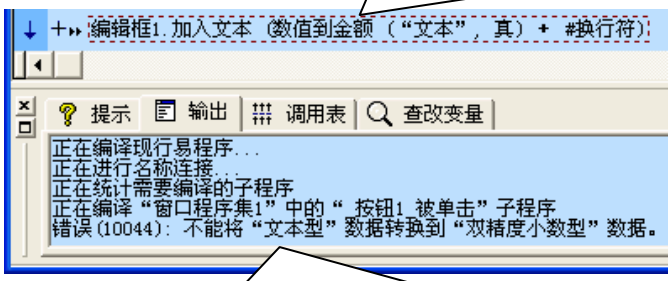
```
容器1 = 100.38
编辑框1.加入文本 (数值到 (变量1, 真) + #换行符)
编辑框1.加入文本 (数值到金额 (四舍五入 (变量1, 1), 假) + #换行符)
```



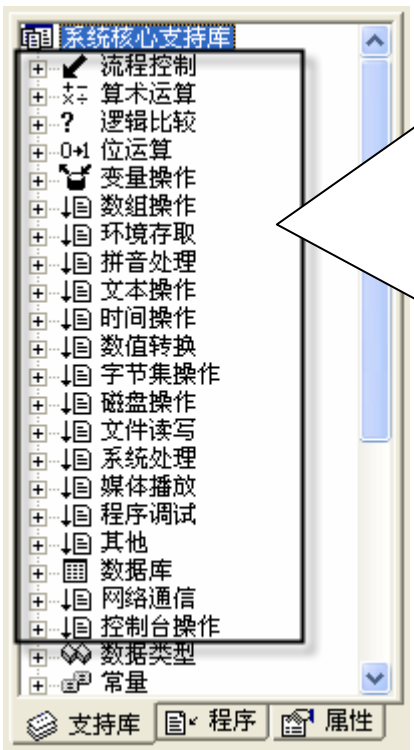
⑨按快捷键 F5 键，试运行这个程序，点击按钮，可以看到运行的结果。
查看完成后，请结束程序试运行。

从“数值到人民币（变量 1，真）”语句可以看出，命令的参数数据可以通过变量来提供。而“数值到人民币（四舍五入（变量 1，1），假）”说明，命令的参数数据还可以通过另外一个命令的返回值来提供。

命令参数类型提供时要类型一致，例如是小数型就不能提供文本型。大家可以试一下，将“数值到人民币（变量1，真）”改为“数值到人民币（"文本"，真）”。



改好后，再试运行一下，就会发现程序不能正常运行，并且光标条会停留在出错行上，输出框中显示程序运行时的错误信息。



易语言中内置了500多种命令，可在任何时间任何地点随意调用。利用它们能够轻松地完成众多复杂的功能。一个复杂的程序通常是由许多命令组合而成。

提示：在**支持库面板**中，所有的系统库函数被分为20类依次列出。单击某个分类前的“+”号使其变为“-”号，即可查看该分类中的命令。而点击其中的任意一个命令名称，立刻就可以在**提示面板**中看到关于该命令的详细帮助。这是在易语言中寻求帮助的很重要的方法。

大家应尽量抽时间多浏览浏览这些命令，只要平时有了一些印象，用的时候再查找就快捷多了。这样在编程时，要实现什么功能，虽有时不能立刻准确地记起要用到的函数，也总能很快地在支持库中查到它。

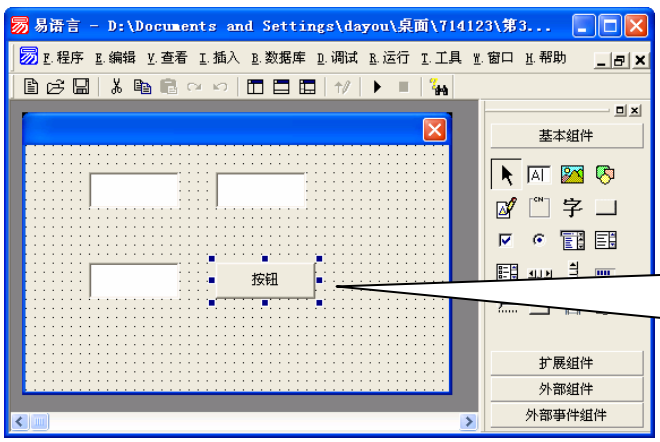
3.2 大小数问题与判断命令

在这里，我们将对两个数字进行判断，以找出其中的最大数。

接下来会详细分析一下易语言的判断命令。通过全可视化设计界面，我们可以非常清楚的了解程序的走向。

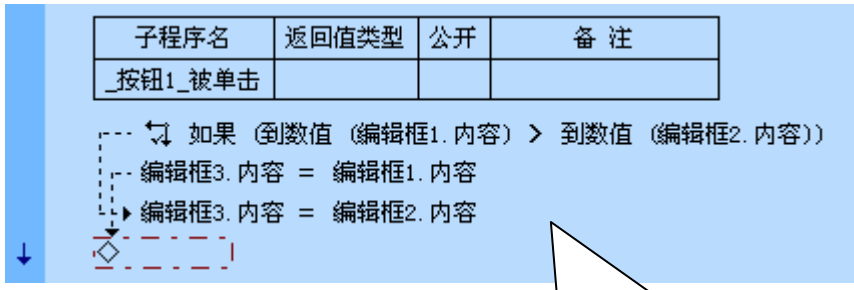
“易语言”中的流程控制类命令目前有下面几种，请先在系统中查看有关各命令的详细解释（指在支持库面板中找到命令后按 F1 即时帮助键查看）。

- 分支类： 如果、如果真、判断
- 循环类： 判断循环首、循环判断首、计次循环首、变量循环首
- 跳转类： 到循环尾、跳出循环、返回、结束

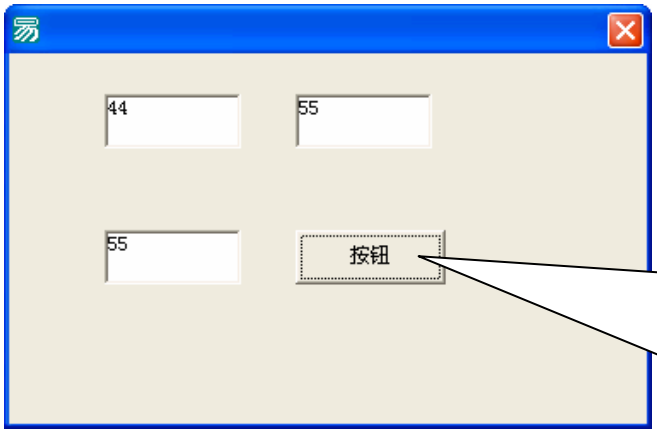


①新建一个易程序，并放三个编辑框，一个按钮在启动窗口中。

②双击按钮，以进入“_按钮 1_被单击”事件子程序的设计界面。



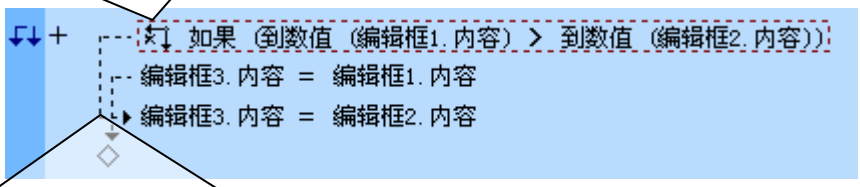
③依次输入以上三行程序代码：



④按 F5 键试运行。在上一排两个编辑框中分别输入两个不同的数字，点击按钮后，就会在下排的编辑框中显示最大的那个数字。

试运行结束后，请退出程序运行。

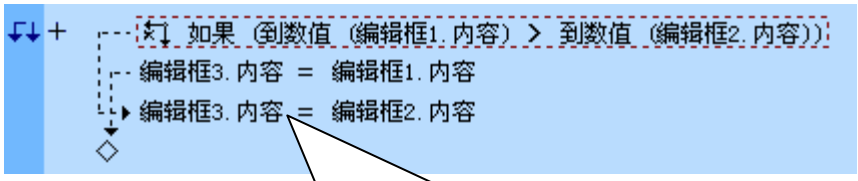
大家将光标定位在如果命令行上，观察一下，就可以发现，标记会在 与 两者之间切换。



与 流程线互相配合。

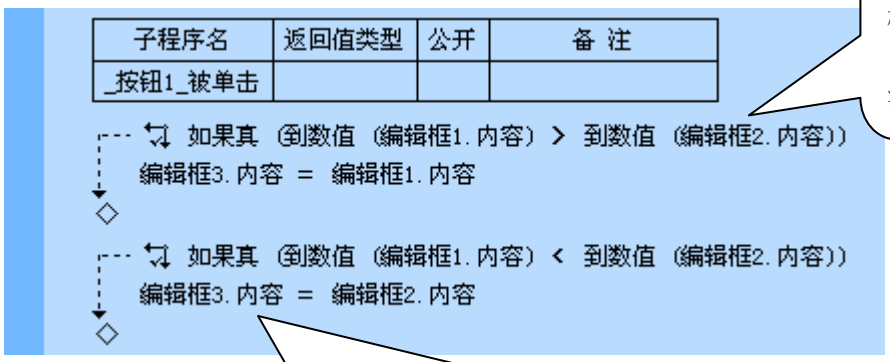
表示当条件成立时，就执行下面的程序。另有一个跳出判断的箭头。

表示当条件不成立时，就执行左边箭头所指向的程序。



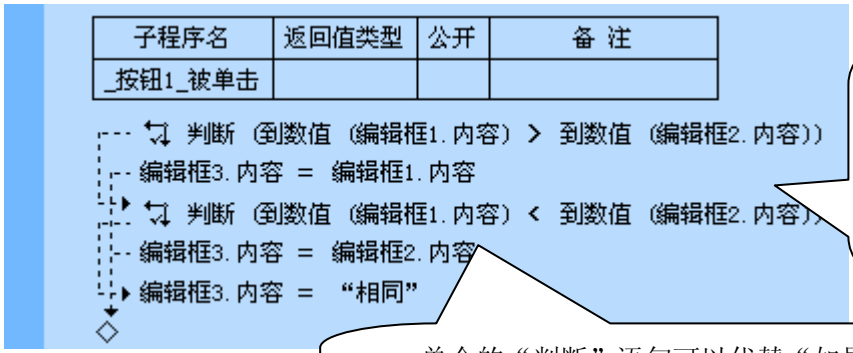
这三行程序代码表示的是：如果编辑框 1 比编辑框 2 大，就在编辑框 3 中显示编辑框 1 的内容，否则就在编辑框 3 中显示编辑框 2 中的内容。

请将上述程序代码删除，改为“如果真”的流程语句。

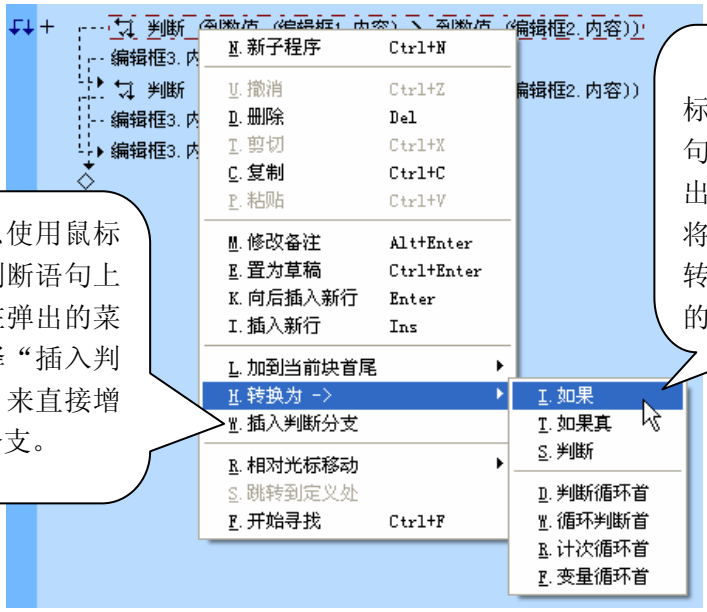


可以看到“如果真”命令与“如果”命令相比少了一个箭头。原来“如果真”命令的条件成立时，即执行条件成立的语句，否则什么也不作。

请将上述程序代码删除，改为“判断”的流程语句。



单个的“判断”语句可以代替“如果”语句。多个判断语句进行判断时是进行同一时间的判断，并且最后有一个默认判断分支。



可以使用鼠标右键在判断语句上单击。在弹出的菜单中选择“插入判断分支”来直接增加判断分支。

可以使用鼠标右键在判断语句上单击。在弹出的菜单中可以将判断语句进行转换到其它类型的。

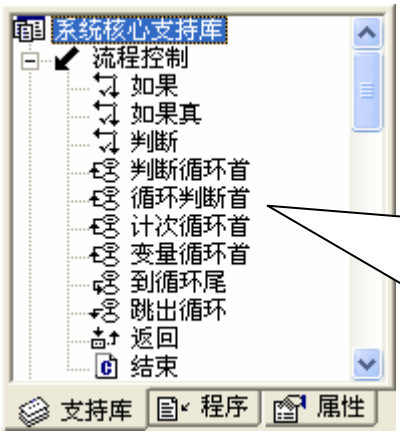
子程序名	返回值类型	公开	备 注
_按钮1_被单击			

编辑框3.内容 = 选择 (到数值 (编辑框1.内容) > 到数值 (编辑框2.内容), 编辑框1.内容, 编辑框2.内容)

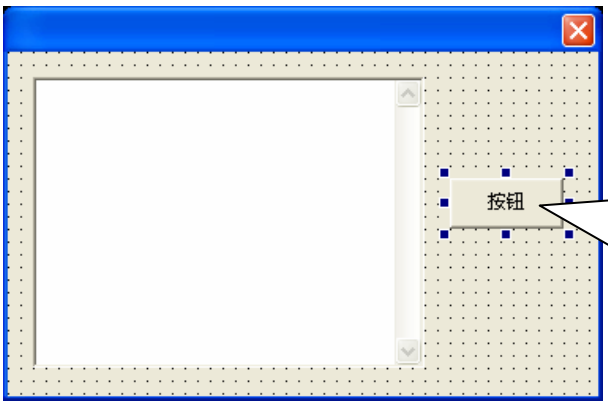
简单的判断也可以用“选择（）”命令代替。
选择命令的第一个参数是成立条件，第二个参数在条件为真时返回本项。第三个参数在条件为假时返回此项。

3.3 循环类命令

前面讲的是判断型命令，在这里要讨论的是循环类命令。



循环类命令有四个，分别是：判断循环首（）、循环判断首（）、计次循环首（）、变量循环首。如果记不住，可以在支持库面板中找到。



如本章开始一样，制作这样一个程序界面。

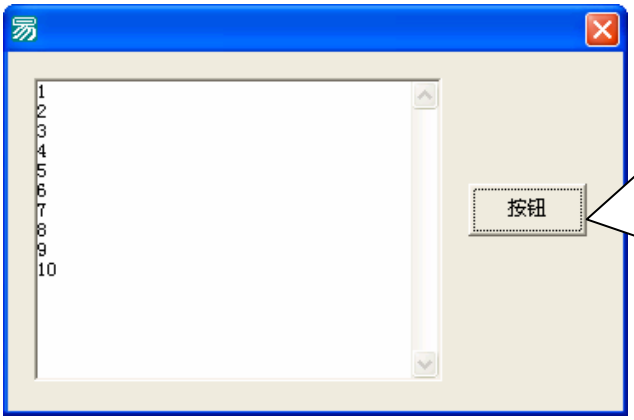
双击按钮控件，进入程序代码录入界面。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	整数型			

```
判断循环首 (变量1 ≠ 10)
    变量1 = 变量1 + 1
    编辑框1.加入文本 (到文本 (变量1) + #换行符)
判断循环尾 ()
```

新增加一个变量，变量名为“变量1”，类型为整数型。再输入四行程序代码。

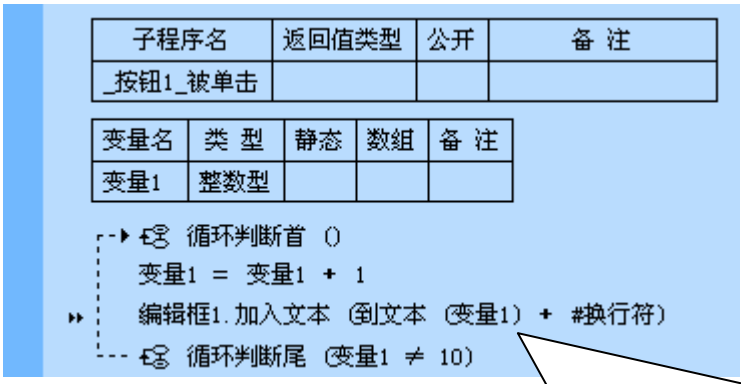


按下 F5 快捷键，试运行这个程序，并且点击其中的按钮，可以看到编辑框中依次显示从 1 到 10 的数字。

原来上述 4 行的对应含意是：

```
判断循环首（变量1 ＝ 10）          //当变量 1 不为 10 时即进行循环
    变量1 ＝ 变量1 ＋ 1             //变量 1 累计加 1
    编辑框1.加入文本（到文本（变量1）+#换行符）  //在编辑框 1 中显示变量 1 的内容
判断循环尾（）                     //返回循环首
```

通过以上即可以实现循环显示 1 到 10 了。



子程序名	返回值类型	公开	备 注
_按钮1_被单击			

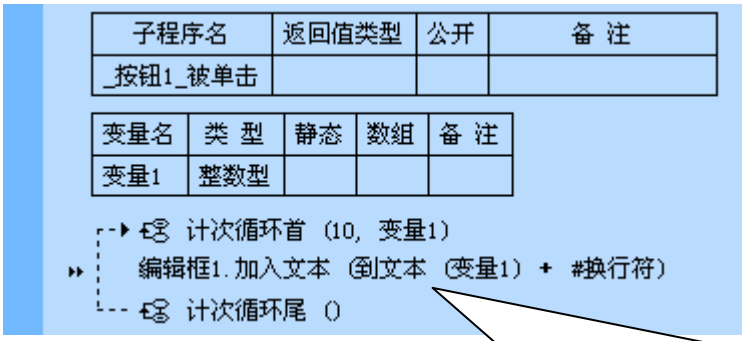
变量名	类 型	静态	数组	备 注
变量1	整数型			

```
循环判断首 ()
    变量1 = 变量1 + 1
    编辑框1.加入文本 (到文本 (变量1) + #换行符)
循环判断尾 (变量1 ≠ 10)
```

“判断循环首”是先判断再循环，而“循环判断首”是先循环再判断。所以两者是有区别的。下面将上述例子中的程序删除，输入以下语句：

```
循环判断首（）                     //循环开始
    变量1 ＝变量1 ＋ 1             //变量 1 累计加 1
    编辑框1.加入文本（到文本（变量1）+#换行符）  //在编辑框 1 中显示变量 1 的内容
循环判断尾（容器1 ≠ 10）           //当变量 1 不为 10 时即进行返回循环首
```

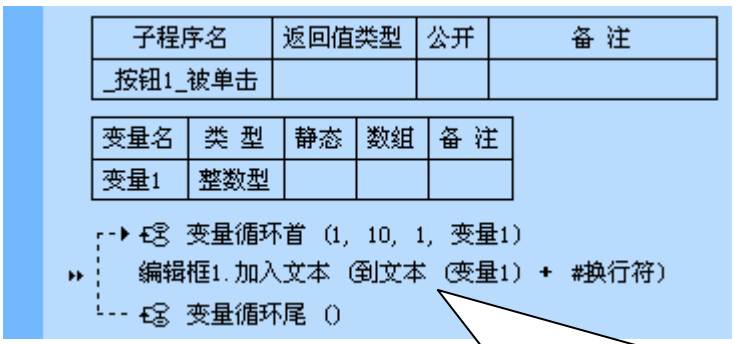
运行后，效果一样，也可以循环显示从 1 到 10。



重新输入以下程序：

```
计次循环首（10，变量 1）                    计次循环开始，变量 1 累加到 10
   编辑框 1.加入文本（到文本（变量 1）+#换行符）   在编辑框 1 中显示变量 1 的内容
计次循环尾（）                              返回循环首
```

运行后，效果一样，也可以循环显示从 1 到 10。



试着将上述程序改成以下：

```
变量循环首（1，10，1，变量 1）  //循环，从 1 开始，到 10 结束，步进为 1，存入变量 1
   编辑框 1.加入文本（到文本（变量 1）+#换行符）  //在编辑框 1 中显示变量 1 的内容
容器循环尾（）                          //返回循环首
```

运行后，效果一样，也可以循环显示从 1 到 10。

通过以上四种循环命令，我们都得到了同样的结果，但在实际应用中，只用其中一种即可。

3.5 跳转类流程控制命令

流程跳转在前两节中已有介绍，当条件满足后，就会循环或不循环，有时会在中途回到第一个循环命令，有时也会提前结束。本节即是讨论这些特别的跳转命令：到循环尾、跳出循环、返回、结束。

“返回”命令因与子程序相关，留到以后讲解。“结束”命令结束当前易程序的执行。

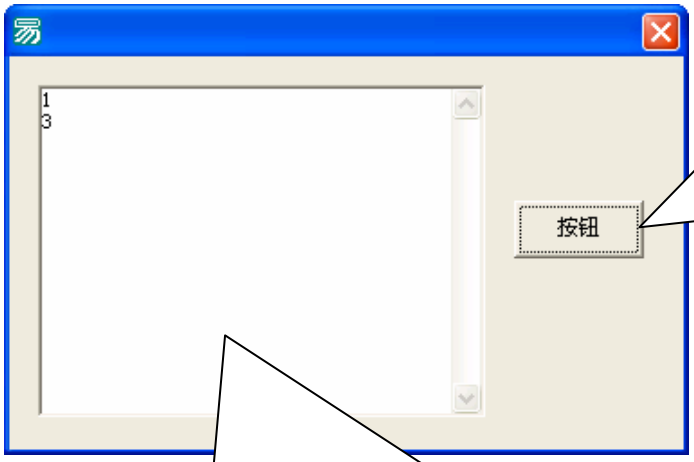
下面结合例程来具体讲述 “到循环尾”、“跳出循环”命令。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	整数型			

```
计次循环首 (5, 变量1)
    如果真 (变量1 = 2)
        到循环尾 ()
    如果真 (变量1 = 4)
        跳出循环 ()
    编辑框1.加入文本 (到文本 (变量1) + #换行符
计次循环尾 ()
```

打开上一节中的例程，将“_按钮 1_被单击”子程序中的所有内容删空，然后在该子程序中重新输入程序代码。



使用快捷键F5试运行。

运行后点击按钮，编辑框中的结果只显示1、3两个数字。

下面来分析为什么会有此结果。

当第1次循环时，变量1的值为1，到循环尾时被显示在编辑框中。

当第2次循环时，变量1的值为2，到第二行如果判断命令时，由于条件成立，被立即跳到循环尾，所以没有显示在编辑框中。又立即跳到循环头，开始新的循环了。

当第3次循环时，变量1的值为3，到循环尾时被显示在编辑框中。

当第4次循环时，变量1的值为4，到第四行如果判断命令时，由于条件成立，被立即跳出了循环，提前结束了循环，所以第五次循环没有，也没有再显示任何内容了。

例子中有一个计次循环，里面有“到循环尾”和“跳出循环”命令。“到循环尾”命令用作跳到当前循环的尾部，即循环尾类命令。“跳出循环”命令用作跳出当前循环。

“返回”是指返回一个值，用于程序的反馈。这个命令多用于子程序。在以后的章节中再细论。

而“结束”语句出现时，即提前结束整个程序的运行。

在上述例子中，即可以将第五行的“跳出循环（）”，改为“结束（）”。大家可以观察一下改过之后的区别。

3.6 易语言常用语句

前面的命令介绍了很多，对于输入程序代码来说，有些语句可以分为四类，大家必须对语句的分类有所了解。

下面列举易语言常见的各类语句供大家学习或编程时参考。

1．值型语句。（也可称属性型语句）

特征：有一个“=”号将左右两边连起来

这是大家学习易语言时首先会接触的一类语句。比如：

```
标签 1.标题 = “汉语言编程技术，易语言！”
```

这句代码的意思是：标签 1 的标题是：“汉语言编程技术，易语言！”——即将标签 1 的标题属性值定为“汉语言编程技术，易语言！”（所谓赋值也）。我们所见的给变量赋值就是用此类语句。赋值语句常见有以下两类：

（1）将某一对象的某种属性值赋给另一对象。比如：

```
标签 1.标题 = 编辑框 5.内容
```

意思即是“标签 1”的标题跟编辑框 5 中的内容一样。比如我们在编辑框 5 中输入“易语言使英语盲也学会了编程”，那么在相关事件（比如单击按钮）的驱动下，标签 1 的标题也相应显示为“易语言使英语盲也学会了编程”。

（2）将某一类型的属性值赋予某个对象。比如：

```
标签 1.标题 = “汉语言编程技术，易语言！”
```

将“汉语言编程技术，易语言！”赋给标签 1 的标题。

所赋予的属性值可以是各种类型的，比如：

```
窗口 1.可视 = 真
```

这个“真”是一个逻辑型数值。如果是文本型数值要用双引号，比如刚才的例子：“汉语言编程技术，易语言！”即是。

又如，我们会发现这类句子：

```
标签 2.标题 = 到文本 (取小时 (取现行时间 ())) + “:” + 到文本 (取分钟 (取现行时间 ())) + “:” + 到文本 (取秒 (取现行时间 ()))
```

因为标签类对象只接受文本型数据，所以要把时间型数据转化为文本型，否则测试时会提示：“所接受的数据类型与传递给它的数据类型不一致”。

有时我们会发现此类句子：

```
变量 1 = 变量 1+10
```

这个语句的意思是：“将变量 1 加 10 之后，再将新值传回给变量 1”。 从这里可以看出，这里的“=”号跟数学中的“=”号在含义上是不同的，在数学中不可能出现 b=b+2 之类的表达式，而在易语言中这种表达式却是允许的，而且是经常运用的。

另外，我们又会看到这类句子：

```
编辑框 1.高度 = 取用户区高度 ( )
```

上述程序可以理解为取得“取用户区高度 （）”的返回值（运算结果），然后将此值赋给编辑框 1 的高度属性。

“=”号的右边表示操控程序的命令，也即是说有时可以将系统命令、执行条件、

2．非运行语句。

非运行语句包括以下几种。

（1）注释型语句

易语言的注释型语句的格式是：

```
※ 注释 注释语句内容
```

注释语句不能执行程序，只是用来解释上一行代码的意思。编译时易语言不会把注释代码也编译到可执行文件中。显示某行代码的注释语句的方法是：选中某行代码，如果在该行代码前出现“+”号，说明该行代码有注解，点击该“+”号可以打开该行代码的注解，点击“-”可以重新隐藏注解。

上述注释型语句是易程序中固有的，我们也可以自己添加注释型语句。方法：直接改动某一备注或用鼠标右键选“修改备注”。

除了上述的备注方法外，在下方的“提示夹”里有对该行代码更详细的解释。

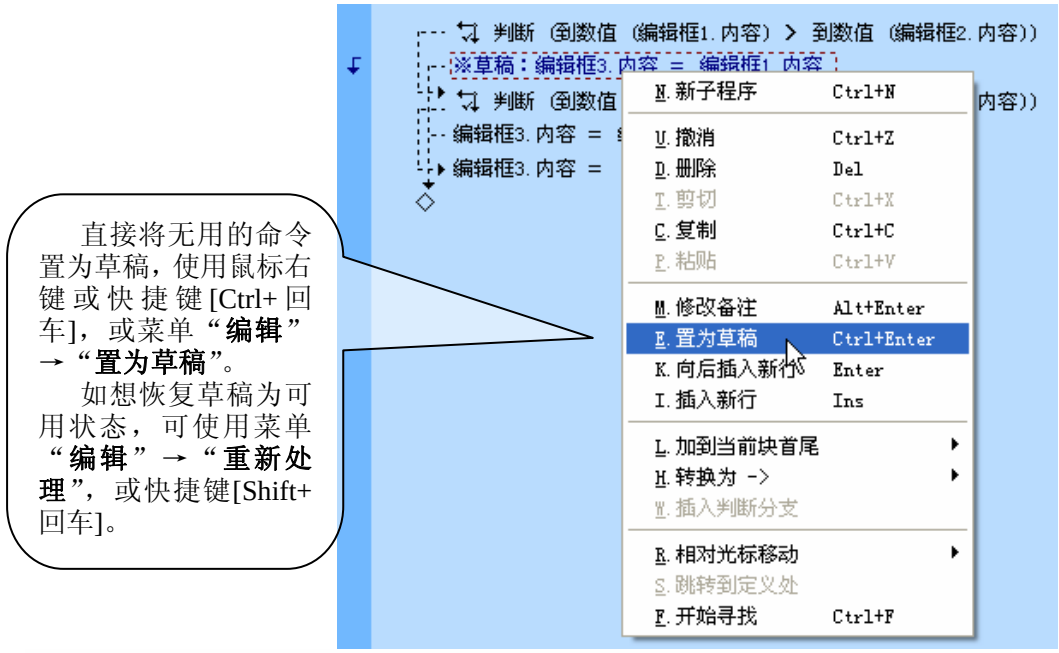
我们也可以把下面的草稿型语句看作为备注。

（2）草稿型语句。

易语言的草稿型语句的格式是：

```
※草稿：程序代码
```

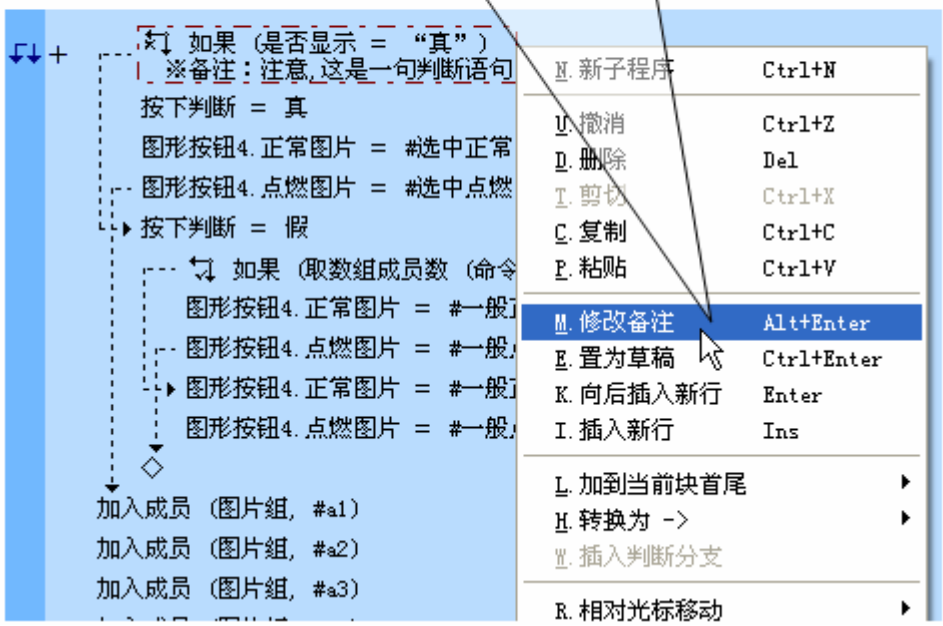
草稿型语句也不能被程序执行，且在编译程序时也不会被编译成机器码。



直接将无用的命令置为草稿，使用鼠标右键或快捷键[Ctrl+回车]，或菜单“**编辑**”→“**置为草稿**”。

如想恢复草稿为可用状态，可使用菜单“**编辑**”→“**重新处理**”，或快捷键[Shift+回车]。

为了增加程序的可读性，可以为每行程序加上备注功能，使用鼠标右键弹出下拉菜单，或使用快捷键[Alt+回车]，再输入备注文字。



3．方法型语句。

方法是一个具体对象能够执行的动作。有的方法会有参数，有的方法则不用参数，“参数”大概相当于调用这个方法的各种相关数据，包括相关对象属性值、系统命令、执行条件、项目、常量、子程序、函数乃至其他对象的方法等，都是可以调用的参数。不同的方法有不同的参数，一种方法可以有多种类型的参数。执行、调用一个对象的方法的一般格式如下：

对象名.方法名(参数 1,参数 2,...)

例如需要在一个名为“购物篮”的列表框里添加一个叫“苹果”的列表项目，其语句如下：

购物篮.加入项目（苹果）

上面这句话的意思可以理解为：“将购物篮的加入项目（方法）定为苹果”。在这个例句中，“加入项目”是“购物篮”的方法，“苹果”是其参数。其中调用列表框的“加入项目”方法的句式是：

列表框.加入项目 (欲加入项目的文本,[与欲加入项目相关的数值])

同时我们又会发现另一类句式，比如：

销毁()

这个句式跟命令型句式很象，但其实它是下面这个句式的省略表达：

控件.销毁 （ ）

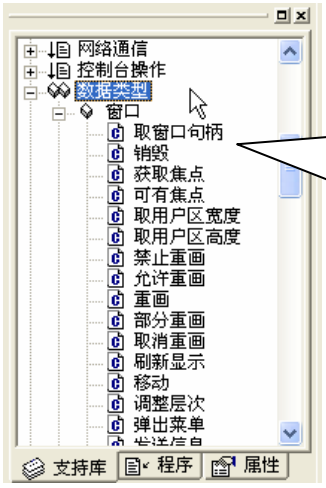
也即是说调用当前控件的方法句式可省略前面的对象名前缀，它仍然是一种方法型语句。

再举另一例子：

控件.弹出菜单 （欲弹出的菜单 ,[水平显示位置],[垂直显示位置]）

可以省略表达成：

弹出菜单 （欲弹出的菜单 ,[水平显示位置],[垂直显示位置]）



欲查看某类控件具有哪些方法、这些方法的详细解释以及有哪些可用参数等，请到易语言设计界面的支持库面板中的“数据类型”分支中寻找。

下面列举几个调用对象方法的例子，以便让大家举一反三：

（1)在程序中使用常量时其名称前必须加上“#”符号,所以如果调用的参数中有常量,就在常量前面加上“#”符号,比如：

```
列表框 1.调整层次 (#底层)
```

（2）调用系统命令作为参数。例句：

```
画板 1.滚动写行 (数值到人民币 (100, 假))
```

这句的意思是，调用“画板 1”的“滚动写行”方法。而其欲写出的数据（参数）是“数值到人民币 (100, 假)”（这句的语法请看帮助)，这是一种数值转换命令语句。

（3）调用相关对象的属性值作为方法的参数。例句：

```
列表框 . 加入项目 (编辑框 1.内容,)
```

其它用法就不一一列举了。

4．命令型语句。

命令型语句跟方法型语句差不多，看看下面两者的对比就知道它们的差别了。方法型语句为：

```
对象名称.方法名称(参数,.....)
```

而命令型语句的基本格式是：

```
命令名称 (参数,.....)
```

“命令”的意思也跟“方法”差不多，都是一种程序运行动作指令，只不过“方法”是某个具体对象所具有的能供调用的指令，而“命令”则是整个易语言系统固有的、可以供任何程序任何对象调用的指令，所以命令名称前面没有前缀的对象名。

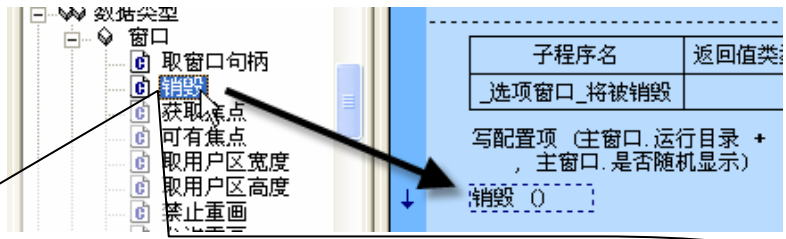
有时候命令的参数不是必需的（即“可选的”），比如：

```
取随机数 ([ 欲取随机数的最小值 ],[ 欲取随机数的最大值 ])
```

“[ ]”表示该参数不是必需的。如果是这种情况，该参数可为空。

比如：

```
取随机数 ( )
```



命令型语句的快速输入方法：在支持库面板中，双击某一个命令或方法，可快速将此命令或方法粘贴到程序行中。

下面再列举一些常见的命令以便举一反三：

（1）载入窗口命令。我们想通过一个窗口启动其它窗口，可用此命令，其句式是：

```
载入 （欲载入的窗口，[父窗口]，是否采用对话框方式）
```

它的意思请查看有关说明。

（2）运行命令。使本程序或其他程序运行，其句式为：

```
运行 （欲运行的命令行，是否等待程序运行完毕，[被运行程序窗口显示方式]）。
```

示例：

```
运行 (“C:\windows\notepad.exe”,,假）。
```

（3）信息框命令 。有时我们想使系统反馈指令，比如弹出相关提示，但又找不到有“信息框”这个控件（控件），原来易语言是通过命令调用信息框的（类似的还有输入框命令），其格式是：

```
信息框 （提示信息，按钮，[窗口标题]）
```

（4）关闭命令。这是一种数据库操作命令，当指定数据库操作完毕后，关闭已经被打开的指定数据库，以便清空数据变量，节约系统资源，其句式是：

```
关闭 （[数据库别名或名称]）
```

3.8 课后练习

（1）用“如果”、“如果真”、“判断”分别实现同一个结果。看看哪一种更加方便、容易。思考一下是否因为某些功能实现结果相同，所以其它命令可以不要，只留一个“判断”命令即可？

（2）用循环命令在编辑框内显示从 11 到 20。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	整数型			

```
计次循环首 (10, 变量1)
    编辑框1.加入文本 (到文本 (变量1 + 10) + #换行符)
计次循环尾 ()
```

用计次循环可以实现。

实际上用其它流程控制命令均可实现。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	整数型			

```
变量1 = 10
判断循环首 (变量1 ≠ 20)
    变量1 = 变量1 + 1
    编辑框1.加入文本 (到文本 (变量1) + #换行符)
判断循环尾 ()
```

用判断循环也可实现。

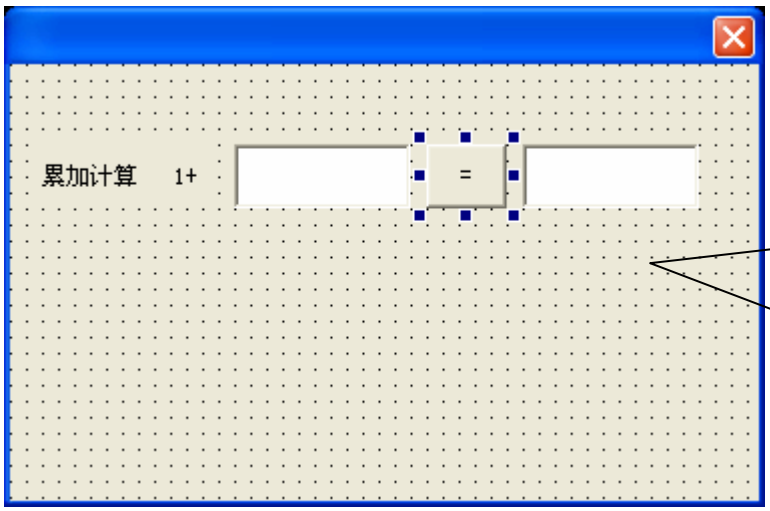
（3）试着将前述程序中的“+ #换行符”删除，运行后观察一下效果。“#换行符”是引用了一个系统常量，将在有关常量的章节中讲述。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	小数型			

```
鸣叫 ()
编辑框1.加入文本 (到文本 (取现行时间 ()) + #换行符)
编辑框1.加入文本 (数值到人民币 (100, 假) + #换行符)
◇
变量1 = 100.38
编辑框1.加入文本 (数值到人民币 (变量1, 真) + #换行符)
编辑框1.加入文本 (数值到人民币 (四舍五入 (变量1, 1), 假) + #换行符)
```

（4）编写一个程序，计算从 1 到某数的累加结果。



新建一个易程序，分别添加标签、编辑框、按钮、编辑框控件。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	整数型			

```
编辑框2.内容 = ""
计次循环首 (到数值 (编辑框1.内容), 变量1)
    编辑框2.内容 = 到文本 (到数值 (编辑框2.内容) + 变量1)
计次循环尾 ()
```

双击按钮组件后输入这些程序代码。最后试运行。

第 4 章 常数、常量与资源

常数、常量、资源用作在程序中提供恒定不变的数据，程序中任何可以使用变量提供数据的地方都可以使用它们。本章将做详细介绍。

本章学习内容：

在前面所讲到的实例中，经常用到赋值给控件的某个属性或赋值给某个变量，例如：**标签 1.标题=“我爱易语言 !”**，“我爱易语言!”就是一个文本常数，当然也可以声明一个文本变量赋值给它，常数可以赋值给属性、变量等。下面我们来认识一下常数的概念。

4.1 常数的概念及分类

常数为可以直接在程序中使用的数据，可以分为以下几种类型：

（1）数值型常数。如：1，16，120，0.15 等。

（2）逻辑型常数。如：真、假。

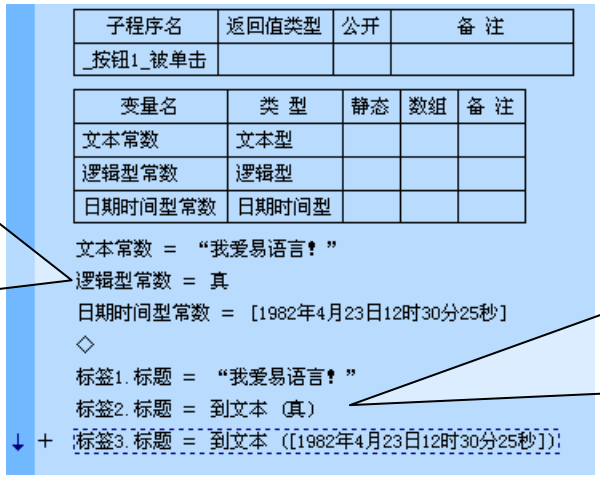
（3）日期时间型常数。日期时间常数的内容必须用中括号括住，并且应按以下格式之一提供，在书写时年份后的时间部分可以被省略：

- [1982 年 4 月 23 日 12 时 30 分 25 秒]；
- [1982/4/23/12/30/25]；
- [1982/4/23/12:30:25]；
- [1982-4-23-12-30-25]；
- [1982-4-23-12:30:25]。

（4）文本常数。文本常数的内容为一段文本，必须使用全角或半角双引号括住。如：″中文编程不是梦！″、“王老师您好!”等等。

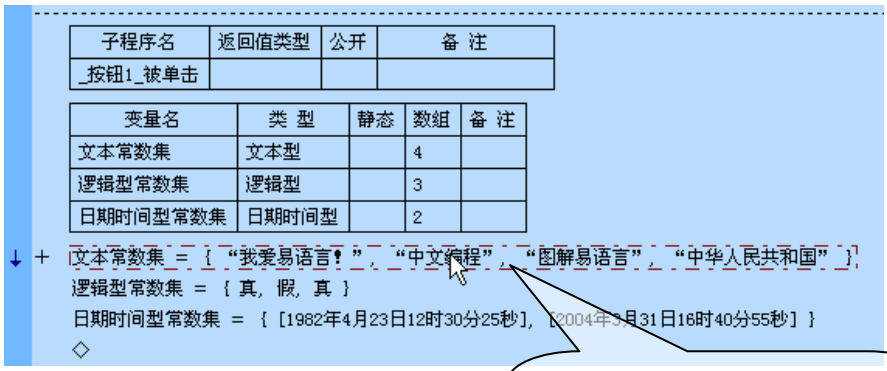
（5）子程序指针常数。子程序指针常数为代表程序中某一子程序的指针数值，表达方式为符号“&”再加上子程序名称。如：“&子程序 1”、“&_启动子程序”等等。

常数赋值给变量。建立变量时注意要与常数数据类型一一对应。



常数赋值给控件属性。如果属性类型不支持赋值的常量，可以通过转换命令改变。

（6）常数集。常数集用作提供各类常数或常量的数组形式。表达方式为使用花括号括住一系列相同类型且通过逗号分隔的各类常数或常量。如：“{1, 2, 3}”、“{"abc", "bcd"}”、“{[1982 年 4 月 23 日], [2000 年 1 月 1 日 12 时]}”、“{&子程序 1, &_启动子程序 }”、“{#红色，#黑色}”、“{#图片 1，#声音 1}” 等等。常数集也可以为空，表达方式为 “{ }”，此时它将被认为包含有 0 个数值型常数，可以用作重新初始化一个数值数组或者将某字节集清空，例如：“数值数组 1 = { }”、“字节集 1 = { }” 等等。



子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
文本常数集	文本型		4	
逻辑型常数集	逻辑型		3	
日期时间型常数集	日期时间型		2	

```
文本常数集 = { “我爱易语言！”, “中文编程”, “图解易语言”, “中华人民共和国” }
逻辑型常数集 = { 真, 假, 真 }
日期时间型常数集 = { [1982年4月23日12时30分25秒], [2004年3月31日16时40分55秒] }
```

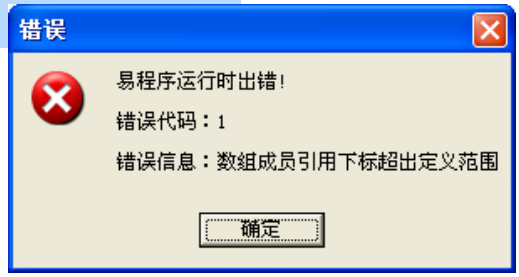
注意：注意常数集里面成员的数据类型必须相同。否者系统会提示出错。

调用常数集的方法很简单，变量名+中括号括住常数对应的位置号，如：标签 1.标题=文本常数集 [1]

```
标签1.标题 = 文本常数集 [1]
```

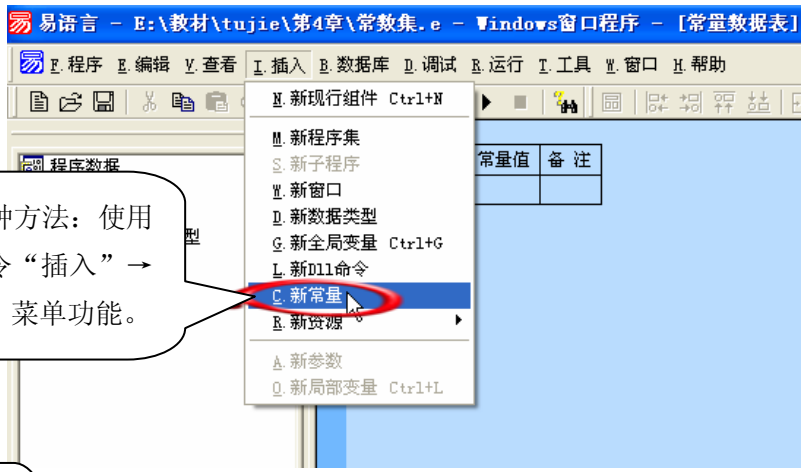
注意：调用常量集里面的成员时，不要超出常量集的成员数，否者编译时会出现提示错误！比如：常量集里面只有 4 个成员，如果读取第 5 个成员，显然是错误的。

```
标签1.标题 = 文本常数集 [5]
```

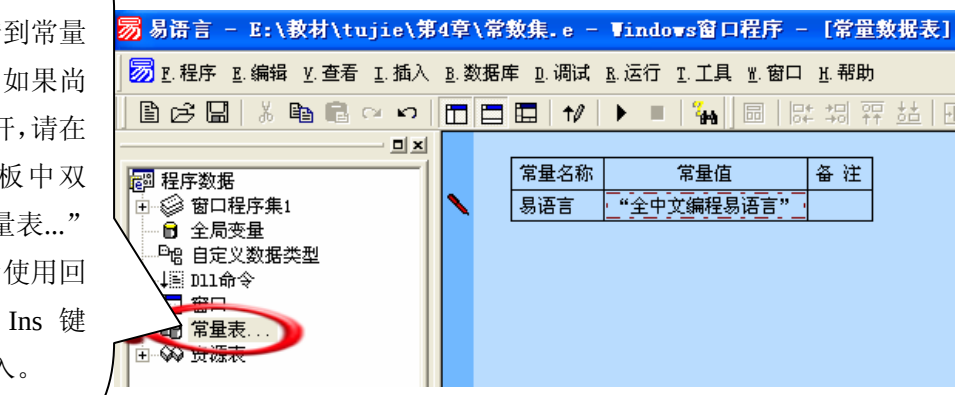


4.2 认识常量

常量用作存储在整个应用程序执行过程中保持不变的数据，其类型可以为数值、文本、逻辑值和日期时间。易语言中已经提供了许多常量，但用户也可以在程序中自行定义。常量的引用表达方式为符号“#”再加上常量名称。如：“#pi”常量等同于数值“3.1415926535”、“#换行符”常量等同于文本 **回车 + 换行** 等等；可以使用以下两种方法之一加入新的空常量。

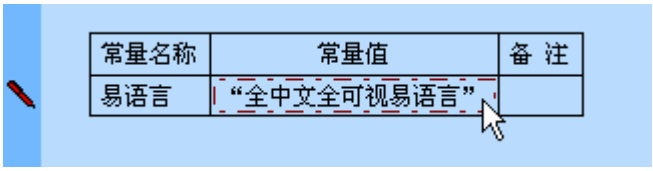


第 1 种方法：使用主菜单命令“插入”→“新常量”菜单功能。



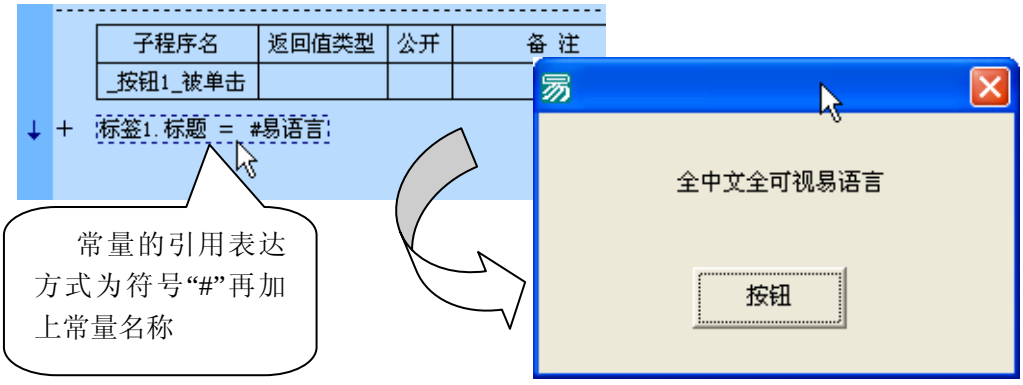
第 2 种方法：跳转到常量数据表，如果尚未被打开，请在程序面板中双击“常量表...”项，然后使用回车或者 Ins 键即可加入。

空常量加入后，可以直接修改其常量名称、常量值、备注等各属性栏。在修改常量值时以 Alt + Enter 键结束输入可以强行将当前输入的所有文本作为文本类型常量内容保存，此方法可用作输入带有双引号的文本常量内容。



4.3 如何使用常量

上面例子中已经建立一个“易语言”常量名，其常量值是“全中文全可视易语言”，下面来练习如何使用这个常量。



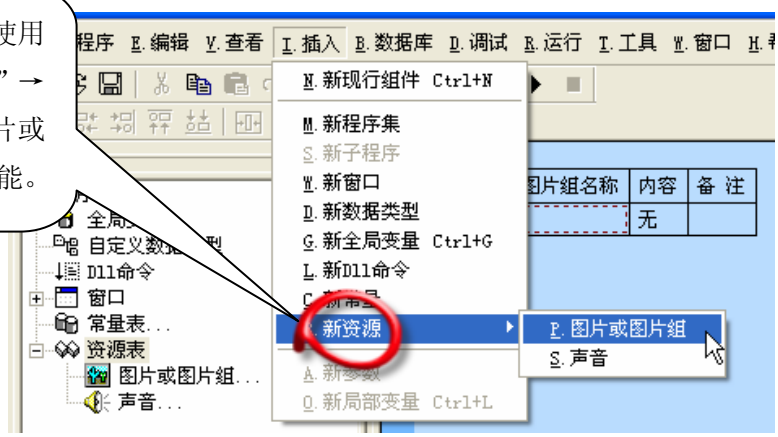
4.4 认识“易语言”资源表

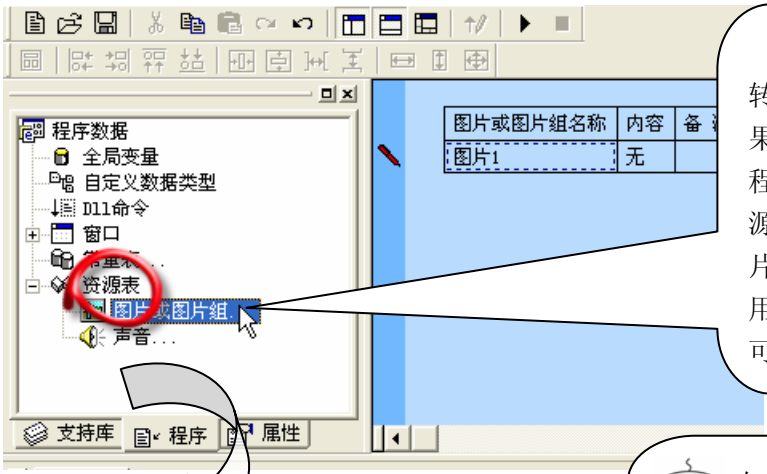
资源用作存储用户需要在程序中使用的图片、声音等数据，其数据类型为**字节集型**，可以被看作为**字节集型常量**，其引用方式等同于常量。

资源资表用来记录在程序中所需要使用的各种资源数据，该数据被直接加入到程序中。

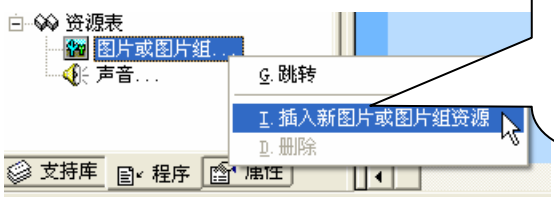
可以使用以下两种方法之一加入新的空图片资源：

第 1 种方法：使用主菜单命令“插入”→“新资源”→“图片或图片组...”菜单功能。



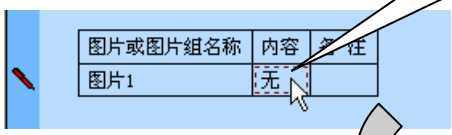


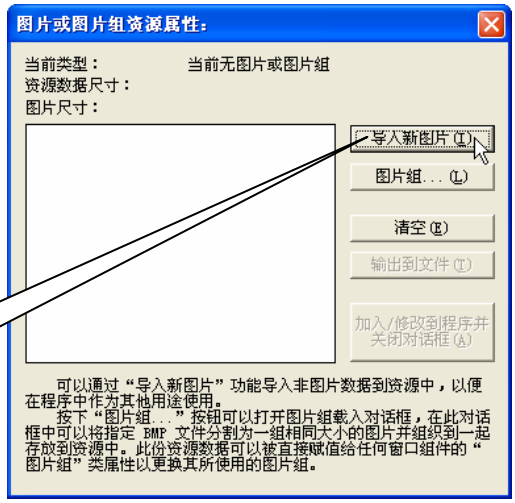
第 2 种方法：跳转到图片资源表，如果尚未被打开，请在程序面板中双击“资源表”→“图片或图片组...”项，然后使用回车或者 Ins 键即可加入。



在“图片或图片组...”上单击右键也可以弹出“插入新图片或图片组资源...”加入。默认文件名为“图片1”，内容为“无”。

单击鼠标，弹出“图片或图片组资源属性”面版。



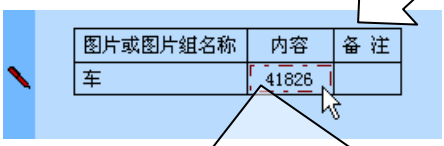


点击“导入新图片”按钮。

选择适合的图片，易语言支持的多种图片格式。

这里显示图片的相关信息。

图片预览窗口。如果不满意，可以重新选择导入。



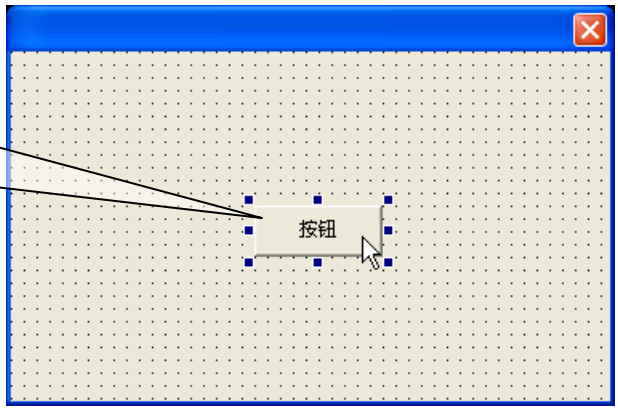
点击“加入/修改到程序并关闭对话框”按钮加入图片资源到程序中。

图片资源导入后，“内容”里面显示的是图片尺寸，单位是字节，把名称更改为“车”。到这里一副图片资源已经导入到程序里面了。

4.5 图片与图标组资源的使用

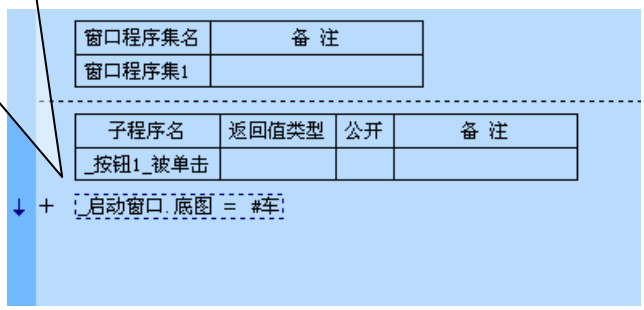
已经知道了怎样把一幅图片加入到程序的资源里面了，那么如何调用它呢？下面练习制作一个简单窗口底图更改的程序。

①新建一个易程序。在窗口中添加一个“按钮”组件。

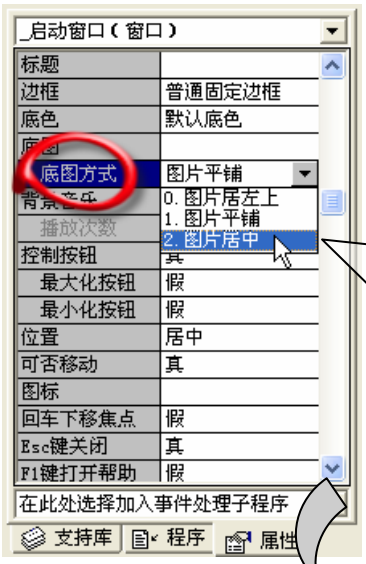


②双击“按钮”组件进入代码编辑窗口，进入“_按钮 1_被单击”事件子程序的代码录入界面。输入以下程序代码：

```
_启动窗口.底图 = #车
```



按 F5 键后试运行。点击按钮，可以观察到窗口的底图变成资源表里面的图像了。

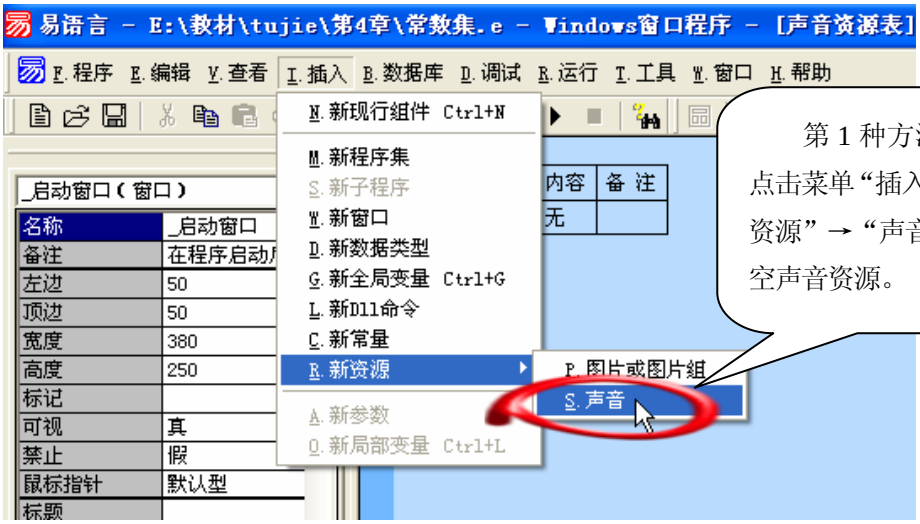


通过上述步骤，大家可以看到，卡车的图形并没有显示完全。可进行以下操作：激活“_启动窗口”，进入属性面板，将“_启动窗口”的“底图方式”属性改为“2.图片居中”。下图为改后试运行的效果。

4.6 声音资源的使用

上节学习了如何加入图片资源，下面再练习加入声音资源。

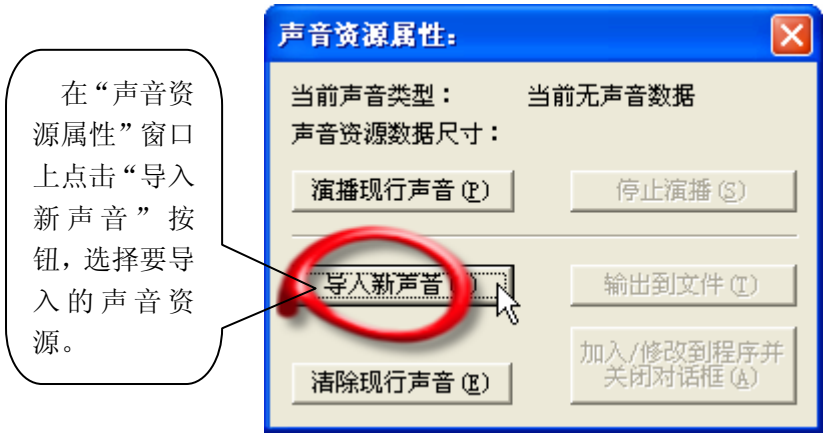
同样也有两种方法加入新的声音资源。



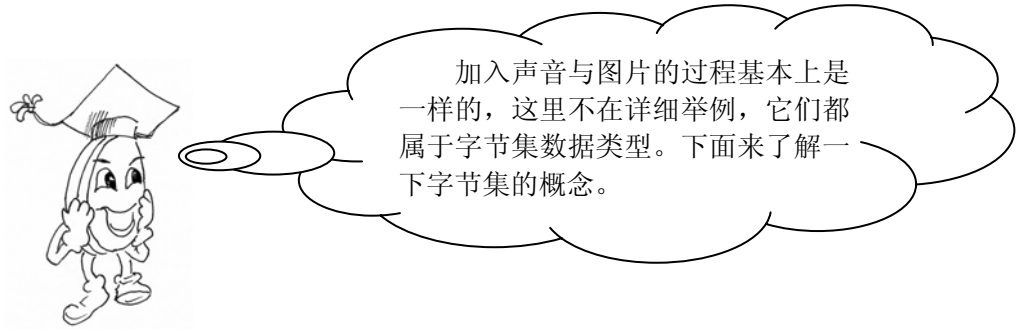
第 1 种方法：依次点击菜单“插入”→“新资源”→“声音”，添加空声音资源。

第 2 种方法：跳转到声音资源表，如果尚未被打开，请在程序面板中双击“资源表”→“声音...”项，然后使用回车或者 Ins 键即可加入。

或使用鼠标右键点击，依次点击“插入”→“新资源”→“声音”菜单添加空声音资源。



4.7 字节集的概念

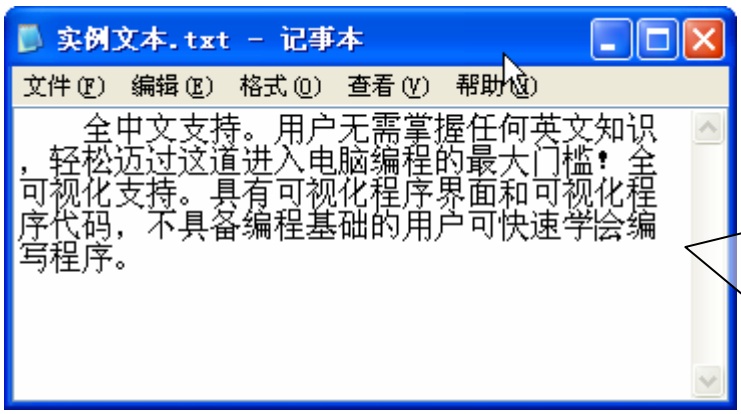


字节集用作记录一段字节型数据。字节集与字节数组非常相似，它们之间可以互相转换。在程序中允许使用字节数组的地方也可以使用字节集，或者相反。字节数组的使用方法，譬如用中括号对（“[ ]”）加索引数值引用字节成员，使用数组型数值数据进行赋值等等，都可以被字节集所使用。两者之间唯一的不同是字节集可以变长，因此可把字节集看作可变长的字节数组。另外，与文本数据一样，多个字节集之间也可以使用相加命令连接为一个字节集。

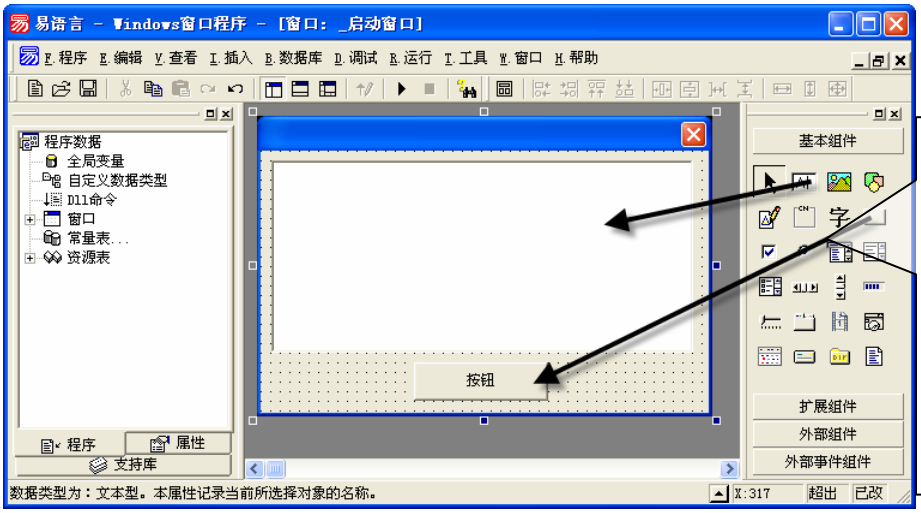
4.8 其他资源的调用

前面所述的是标准的图片资源与声音资源，而本节将介绍资源的其他用法，实际上这些用法在编程中使用还比较多。

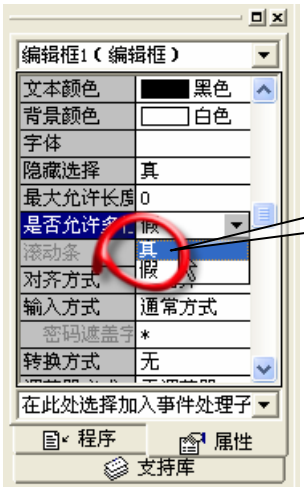
既然资源是字节集格式，能否把一个文本文件当成一个资源调用呢？答案是可以的，下面通过一个程序练习调用文本文件资源。



①启动系统附件中提供的记事本程序，输入一段文字，然后保存文件，作为本例导入的资源。

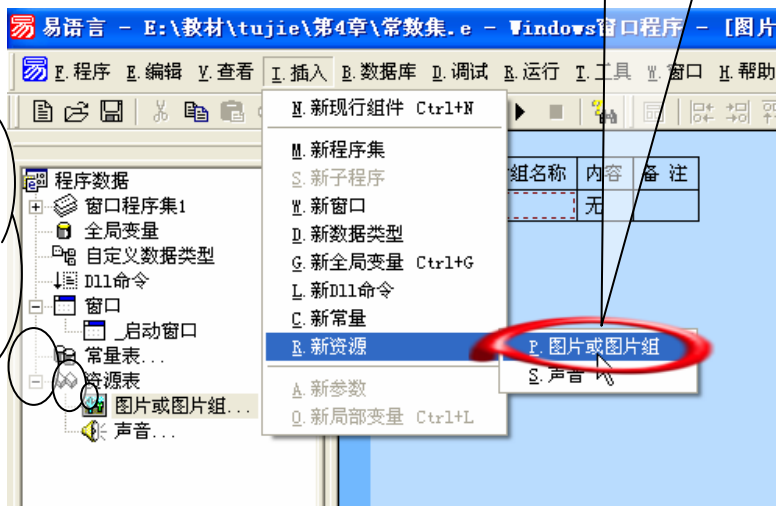


②启动易语言程序，在窗口上添加一个编辑框和一个按钮组件。



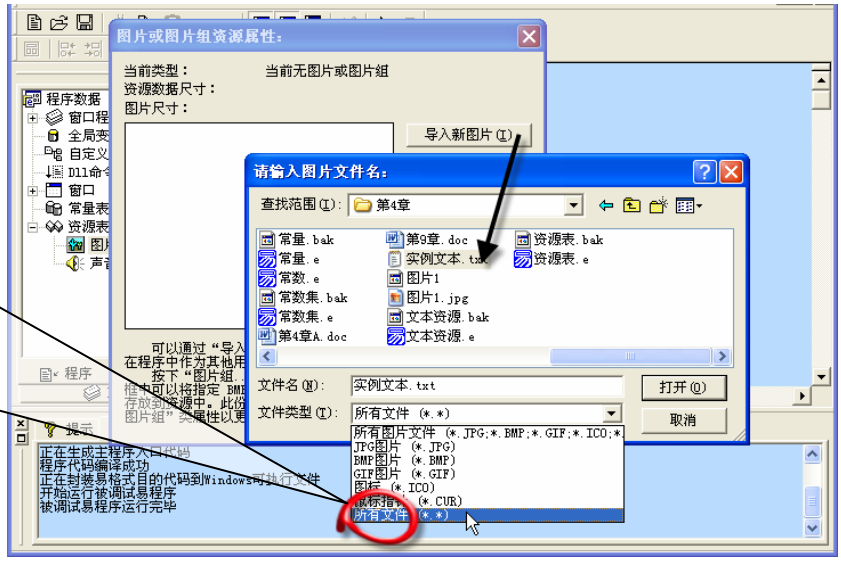
③更改编辑框属性面板上“是否允许多行”为真，这样文本就可以支持多行文本。

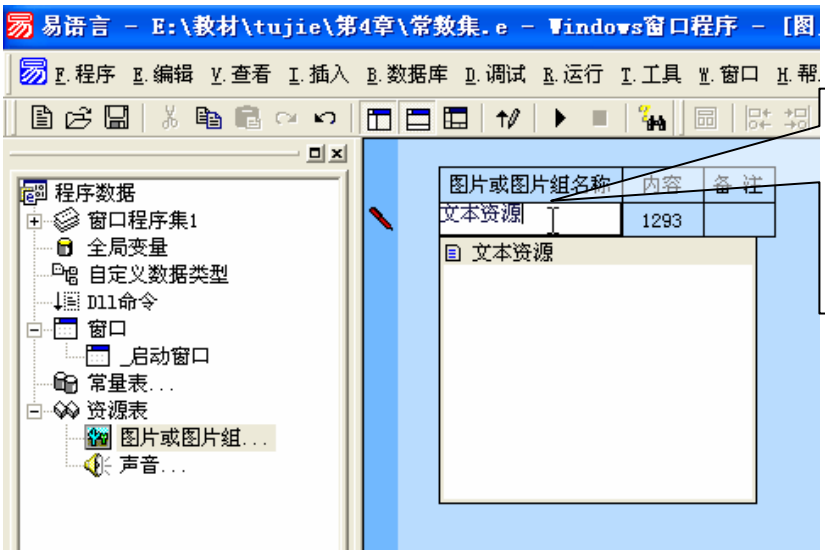
④依次点击“插入”→“新资源”→“图片或图片组”菜单，添加一个空图片资源。



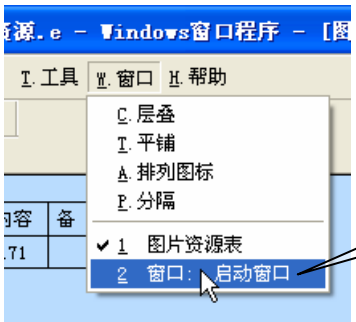
还记得前面说过的插入新图片资源的第二种方法吗？对就是通过“资源表”添加。

⑤在选择新图片资源时，更改打开文件类型为“所有文件”，这样就可以显示所有格式文件，选择刚才保存的文本文件名。





⑥资源导入后，更改文件名为“文本资源”。

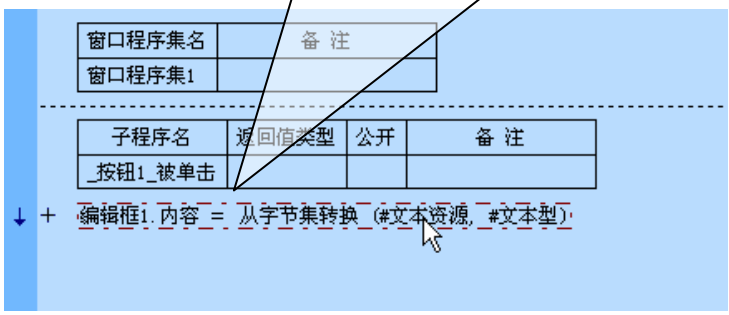


⑦切换到“__启动窗口”窗口面板。

⑧双击按钮组件，进入“_按钮 1_被单击”子程序。输入：

编辑框 1.内容 = 从字节集转换 (#文本资源, #文本型)

编辑框里面显示的是文本数据，所以用命令把字节集转换为文本。



程序制作完毕，按“F5”运行，点击按钮，看看文本资源里的文本是不是显示到编辑框里面了。

4.9 课后练习

（1）判断下面程序代码的正确性，如果错误，请给出正确的表达方式。

① 标签 1.标题=100

② 编辑框 1.内容=“易语言”

③ 字节集变量=｛ 0 ,“中国”, 158 ｝

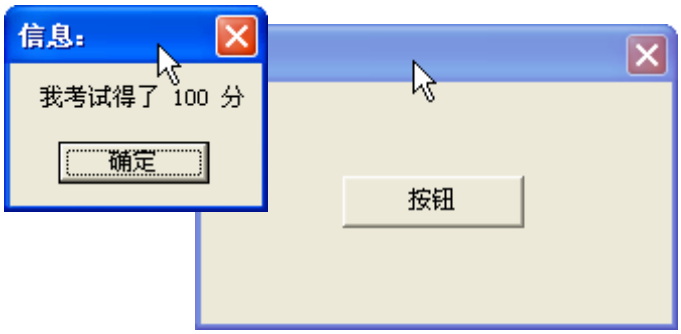
（2）把“我考试得了１００分”分别设为3个常量，分别为：“我考试得了”、“１００”、“分”，利用“信息框（）”命令显示出来。

常量名称	常量值	备 注
常量1	“我考试得了 ”	
常量2	“100”	
常量3	“ 分”	

窗口程序集名	备 注
窗口程序集1	

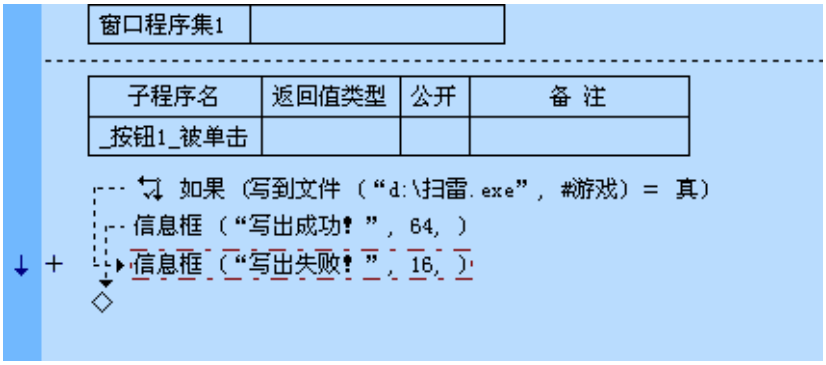
子程序名	返回值类型	公开	备 注
_按钮1_被单击			

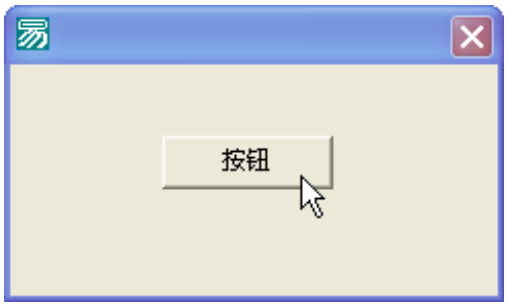
信息框（#常量1 + #常量2 + #常量3, 0, ）



（3）输入或修改当前常量的值时，此值可以为数值、文本、逻辑值或日期时间。提示：以 _____ 结束输入可以强行将当前输入的所有文本作为文本类型常量内容保存，此方法可用作输入带有双引号的文本常量内容。

（4）练习把 Windows 系统自带的“扫雷”游戏程序（winmine.exe），导入到“声音…”资源表里，然后写出文件。





第 5 章 制作菜单

本章主要介绍应用程序菜单的制作，并举出一个记事本的例子，介绍判断语句，以及介绍选择语句和循环语句。

本章学习内容：

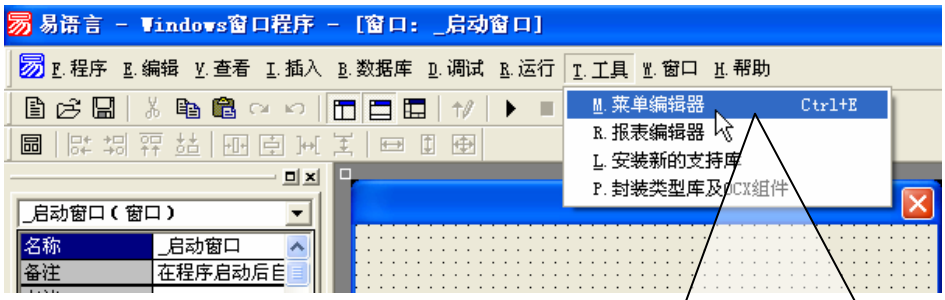
一般应用程序都带有一个组织分工明确的菜单。

制作菜单需要在窗口中使用鼠标右键弹出编辑菜单的命令，当输入菜单内容后，才可以在窗口上方显示菜单，但菜单也有它的事件，也有它的属性，因此，我们也将菜单作为一个控件来介绍。

一般的应用程序都会有“菜单”和“菜单工具栏”，比如易语言的操作界面就有“文件”、“编辑”、“查看”和“插入”等菜单。建立菜单可以精简程序界面。

下面就通过建立一个记事本的菜单来学习吧。

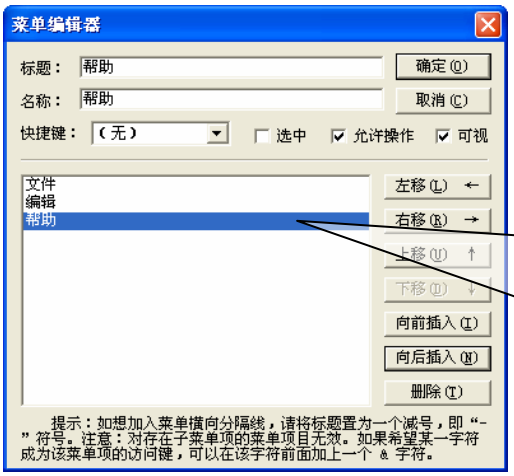
5.1 菜单的简单建立



①可由两种方法打开菜单编辑器。

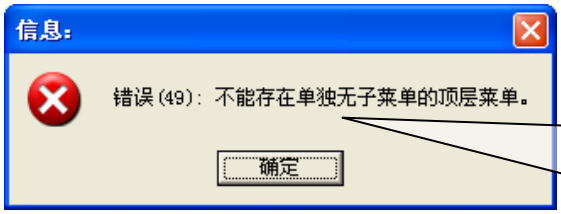
第一种方法是使用菜单“工具”→“菜单编辑器”。或使用快捷键[Ctrl+E]可弹出建立菜单的对话框。

第二种方法是随时在窗体设计时窗体的空白区点击鼠标右键，可弹出下拉菜单，从中选“菜单编辑器”。

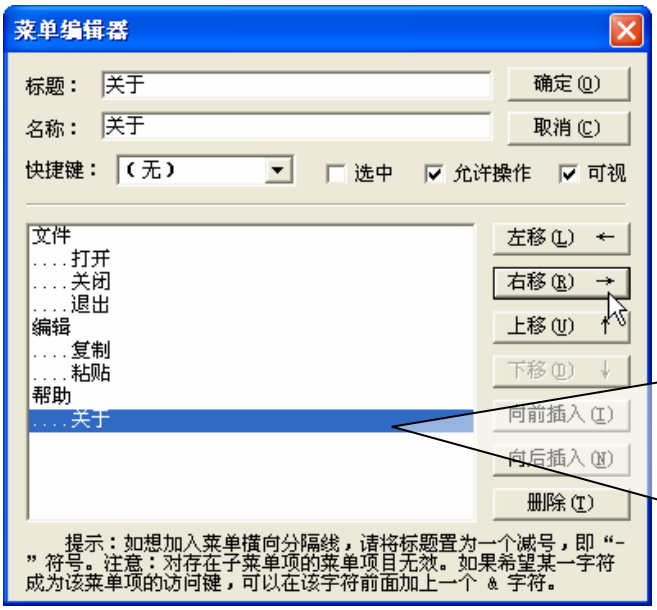


②先建立顶层主菜单。

标题中填入“文件”后，就建立了第一个主菜单项，要建立第二个主菜单项点“向后插入”，继续填入“编辑”、“帮助”，依此类推，建立其余主菜单。



如果这时按下确定钮，会弹出错误提示。这是因为还没有将子菜单建立。因此还要补上子菜单。

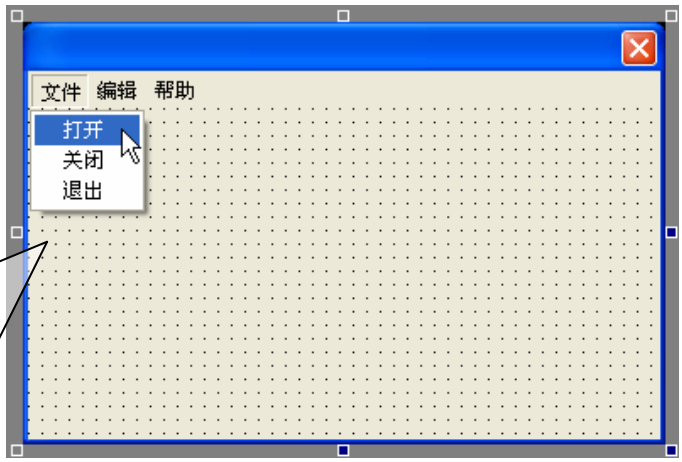


③先按照建立主菜单的方法建立某个菜单项，再点“右移→”按钮，它就会变成上一个菜单的子菜单。子菜单的显示前面有若干点号（“....”号）。

对于二级子菜单、三级子菜单来说，只要多点击几次“右移→”，就可以实现多级菜单了。

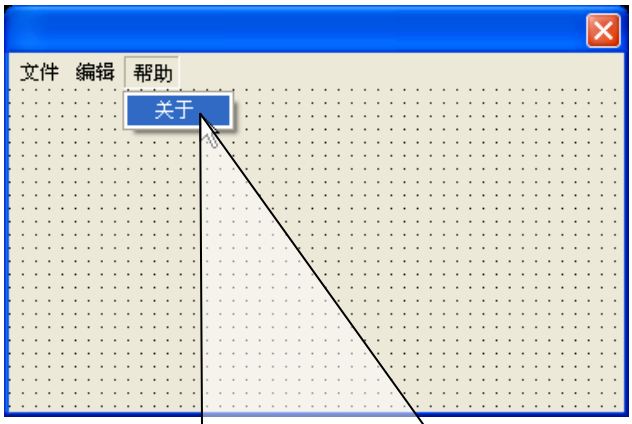
④菜单基本设计好后，点击确定按钮，以回到窗体设计界面，可以观察到菜单已经建立好了，并且点击主菜单，就会向下拉出下级菜单来，直接运行时也是这样。

如没有设计好，还可以回到第一步再改。



菜单设计中的注意事项

1．菜单设计中的“标题”文字可以重复，而“名称”不可以重复。
这是因为标题只是显示在屏幕上供大家看的，而名称是由程序内部引用，类似于按钮控件中的名称属性。只能是唯一的，不能重复。
这个原理与按钮控件一样，按钮控件也是可以标题一样，但名称绝对不能设定为一样的。
2．不能将一些阿拉伯数字放在名称的最前面。
3．“标题”的文字可以和“名称”不一样。
4．系统会自动删除“名称”属性中的空格，在“标题”中可以加入空格，而在“名称”前加入空格就会被自动删除。这是因为在程序的引用中不能有空格。



完成上述后按 F5 键试运行一下，这时点击菜单项后一点反应也没有，这是因为还没有为每个菜单项加入程序代码。

⑤如何为菜单项加入程序代码呢？请取消试运行，回到程序设计状态，可以直接用鼠标选中某一个子菜单项，一松手即会自动进入程序代码设计界面。

例如，用鼠标选中“帮助”→“关于”菜单后点击，会产生“_关于_被选择”的子程序。

子程序名	返回值类型	公开	备 注
_关于_被选择			

信息框（“现在可以显示帮助信息了!”，0，）

⑥现在就输入一行简单的命令，例如输入：

```
信息框（“现在可以显示帮助信息了!”, 0，）
```

运行时，用鼠标选中“帮助”→“关于”菜单后点击，会运行“_关于_被选择”的子程序。而子程序就是弹出一个信息框。

5.2　菜单的属性

在上述菜单的设计中，菜单也有它自己的一些属性，我们可以通过设计时修改这些属性。而在程序运行时，通过程序代码改属性的方法也将在后面章节中讲述。

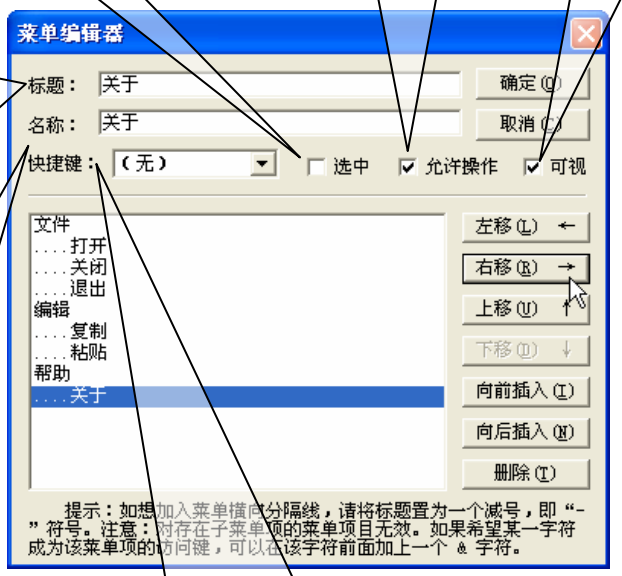
“选中”属性可以控制是否在子菜单前面加勾。

“允许操作”属性可以控制子菜单是否可以操作，在运行时是灰色表示的。

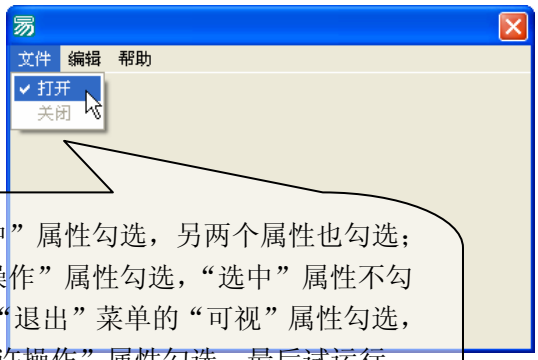
“可视”属性可以控制子菜单是否可以看见。

“标题”属性是菜单显示的文字。可以重复，可以加空格。

“名称”属性是程序内部引用的关键字。不可以重复，不可以加空格，且首字不能为半角字母。

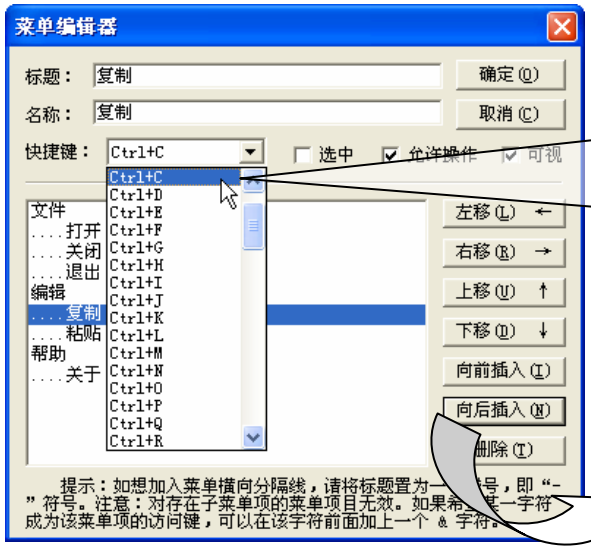


“快捷键”属性可以为每个子菜单项提供热键，通过快捷键执行程序。

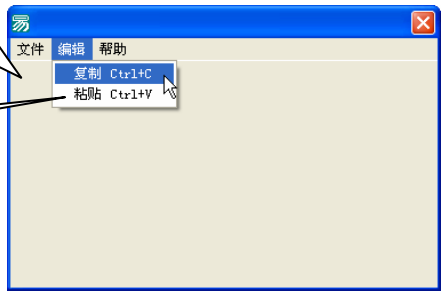


将“打开”菜单的“选中”属性勾选，另两个属性也勾选；将“关闭”菜单的“允许操作”属性勾选，“选中”属性不勾选，“可视”属性勾选；将“退出”菜单的“可视”属性勾选，“选中”属性不勾选，“允许操作”属性勾选。最后试运行，看看运行后的效果。

可以看到菜单中“打开”菜单前有一个勾，“关闭”菜单变灰不可选了，“退出”菜单看不见了。



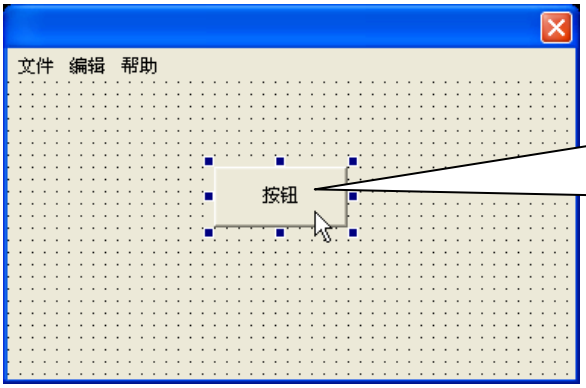
将“复制”菜单的快捷键属性设置为[Ctrl+C]；将“粘贴”菜单的快捷键属性设置为[Ctrl+V]。设置完成后请试运行一下，看看运行后的效果。



试运行后的效果，可以看到菜单后都加入了快捷方式。

5.2 如何引用菜单项编程

上面的菜单设计都是在程序设计状态下进行的，有没有办法可以用程序的方法控制上述的属性，从而让用户在使用中也可以改变菜单的属性呢？下面仅作一个小的试验即可了解如何编程实现了。



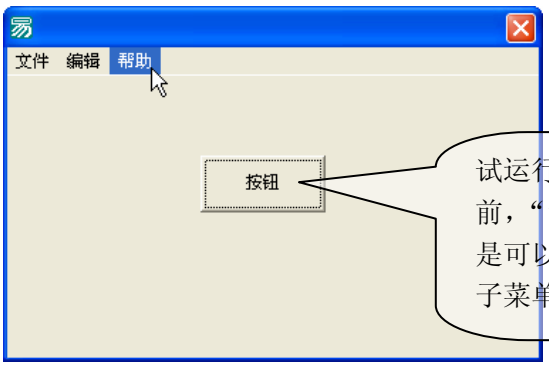
打开前述的例程，新增加一个“按钮”组件，双击此按钮组件，进入程序代码输入界面。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

关于.可视 = 假

在“_按钮 1_被单击”子程序中输入一行程序代码：

```
关于.可视 = 假
```



试运行这个易程序，在没有点击按钮之前，“帮助”菜单中的“关于”子菜单是可以显示的，以点击按钮后，“关于”子菜单就无法显示出来了。

虽然菜单会被隐藏，但还是可以再次显示出来的。只要用以下程序代码即可实现：

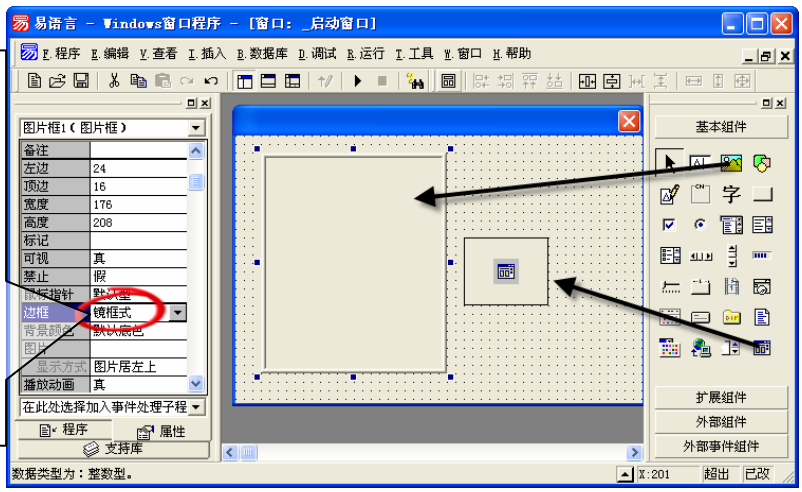
```
关于.可视 = 真
```

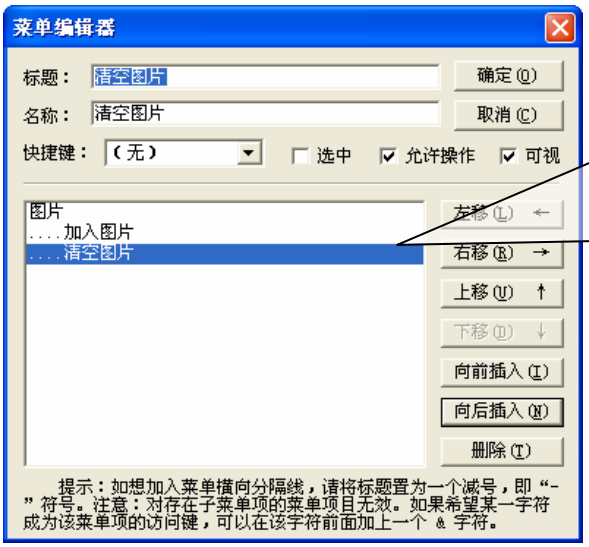
5.3 如何引用菜单项编程

前面只简单的应用了一个按钮控制菜单的属性，而菜单如何控制窗体中的其它控件或执行命令功能呢？这就是本节所要介绍的。

下面跟着例程来学习吧。

①新建一个易程序，在窗口中放一个图片框组件和一个通用对话框组件，并在图片框组件的属性表中将“边框”属性改为“镜框式”。

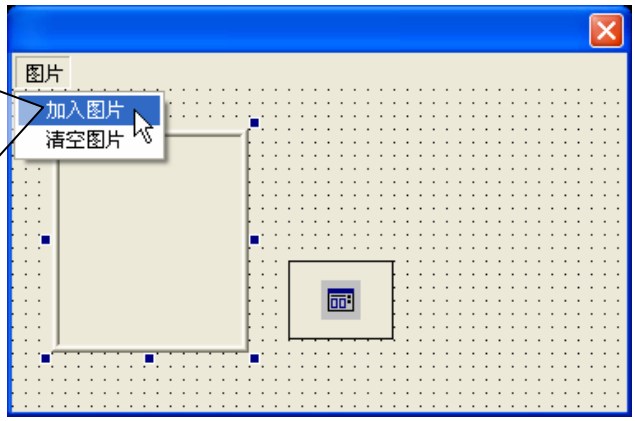


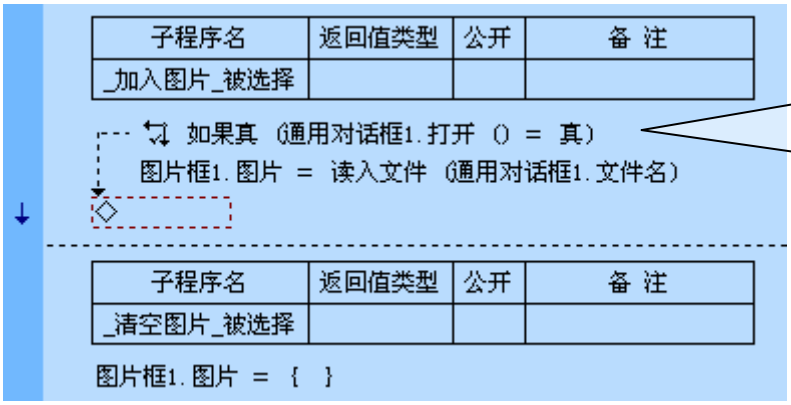


②在窗口中点击鼠标右键，选中“菜单编辑器”选项后，会弹出“菜单编辑器”对话框，建立一个主菜单，主菜单名为“图片”，子菜单为“加入图片”、“清空图片”。

③以上只差加入程序代码，让菜单能够执行了。

在窗口设计中点菜单，可以看到下级子菜单，分别用鼠标点中子菜单就可以进入程序代码输入界面了。





④分别在事件子程序中输入程序代码。

⑤按 F5 键试运行一下，选择“加入图片”菜单就会弹出一个打开图片文件的对话框，从中选一张图片就会显示在图片框中。

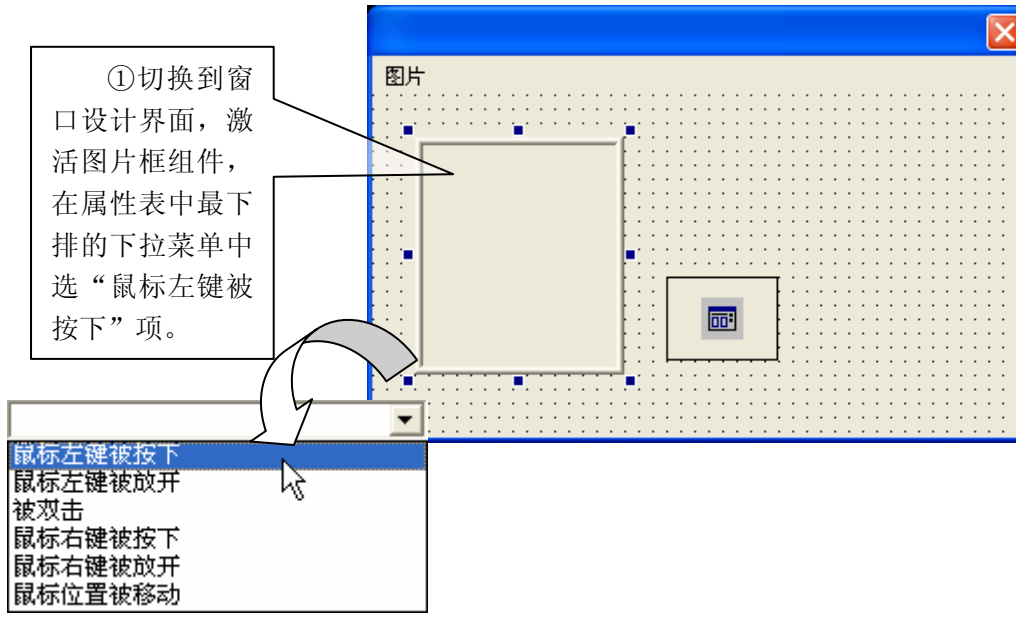
如果选中“清空图片”的子菜单，就会清除图片框中的图片。

5.4 弹出式菜单

前面的菜单是从窗口上部调用的，能不能不用菜单，直接点击图片框，从中弹出一个菜单进行操作呢？

大家跟着本节实例制作的步骤来学习吧。

①切换到窗口设计界面，激活图片框组件，在属性表中最下排的下拉菜单中选“鼠标左键被按下”项。



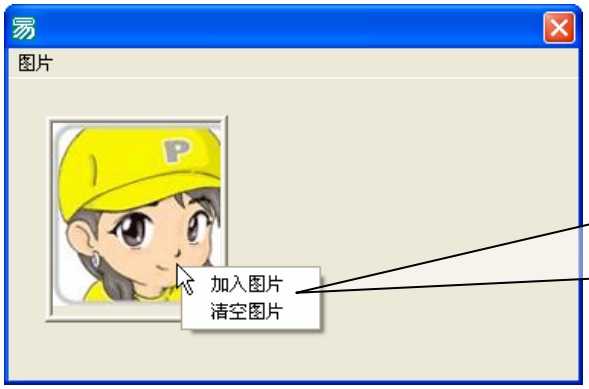
子程序名	返回值类型	公开	备 注		
_图片框1_鼠标左键被按下	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

弹出菜单（图片，，）

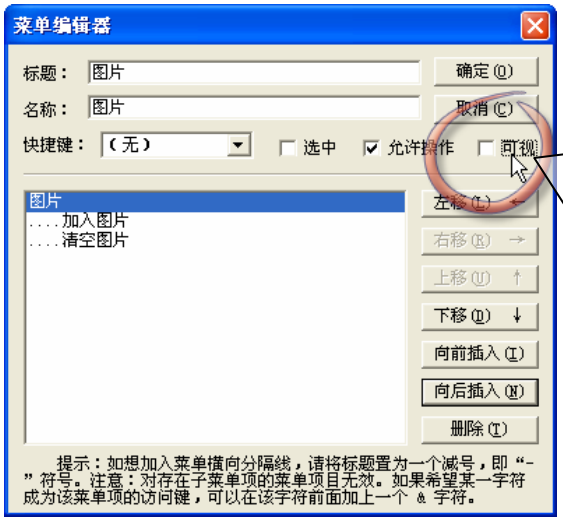
②输入一行程序代码即可。

这句表示弹出名称为“图片”的菜单。

最后试运行一下。



③试运行时，大家可以通过用鼠标左键单击图片框，可以弹出菜单了。



④如果希望在运行时也不显示主菜单于窗口的上部，可以进入菜单编辑器，将“图片”菜单的“可视”属性去除勾选。

去除后再试运行看一看。

⑤试运行后，就看不到“图片”主菜单了。

5.5 定位弹出菜单

前面的菜单直接点击图片框后，从中弹出一个菜单，能不能在鼠标上方弹出呢？

这里再提供一个简单的例程，大家跟着本节实例制作的步骤来学习吧。

子程序名	返回值类型	公开	备 注		
_图片框1_鼠标左键被按下	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

弹出菜单（图片，取鼠标水平位置 () - 10，取鼠标垂直位置 () - 45）

打开前面的例程，将原弹出命令改为以下：

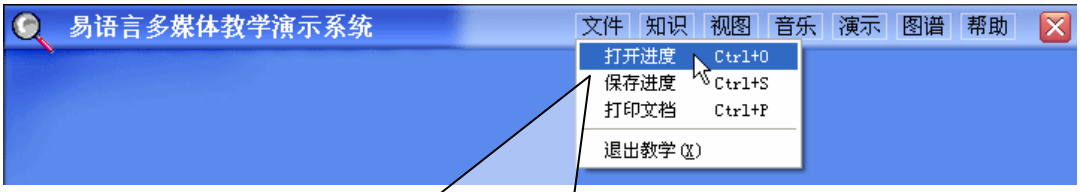
```
弹出菜单（图片，取鼠标水平位置 () - 10，取鼠标垂直位置 () - 45）
```

试运行后可以看到，鼠标在图片框中点击后，会在上面弹出菜单。为什么会这样呢，让我们来分析一下程序代码吧。

使用键盘上的右方向键展开程序代码，可以看到是因为利用了弹出菜单命令的另两个参数，以设计弹出的位置。

子程序名	返回值类型	公开	备 注		
_图片框1_鼠标左键被按下	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

```
弹出菜单 (图片, 取鼠标水平位置 () - 10, 取鼠标垂直位置 () - 45)
    ※欲弹出的菜单： 图片
    ※水平显示位置： 取鼠标水平位置 () - 10
    ※垂直显示位置： 取鼠标垂直位置 () - 45
```



实际上使用这样的菜单显示方式在一些场合还很多。如：使用“图形按钮”组件代替菜单，这时就要在新按钮下方位置显示菜

子程序名	返回值类型	公开	备 注
_图形按钮1_被单击			

```
弹出菜单 (文件, _启动窗口.左边 + 383, _启动窗口.顶边 + 23)
```

子程序名	返回值类型	公开	备 注
_图形按钮2_被单击			

```
弹出菜单 (知识, _启动窗口.左边 + 426, _启动窗口.顶边 + 23)
```

子程序名	返回值类型	公开	备 注
_图形按钮3_被单击			

```
弹出菜单 (视图, _启动窗口.左边 + 469, _启动窗口.顶边 + 23)
```

子程序名	返回值类型	公开	备 注
_图形按钮4_被单击			

```
弹出菜单 (音乐, _启动窗口.左边 + 512, _启动窗口.顶边 + 23)
```

这里是部分程序代码，可以看到为每个图形按钮的被单击事件中写入定位弹出菜单的命令。

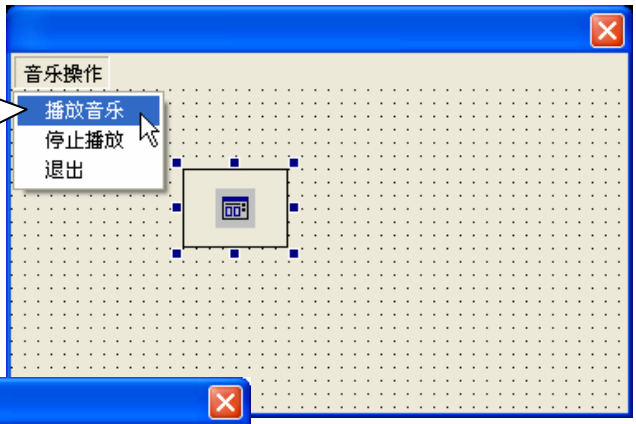
5.6 托盘式菜单

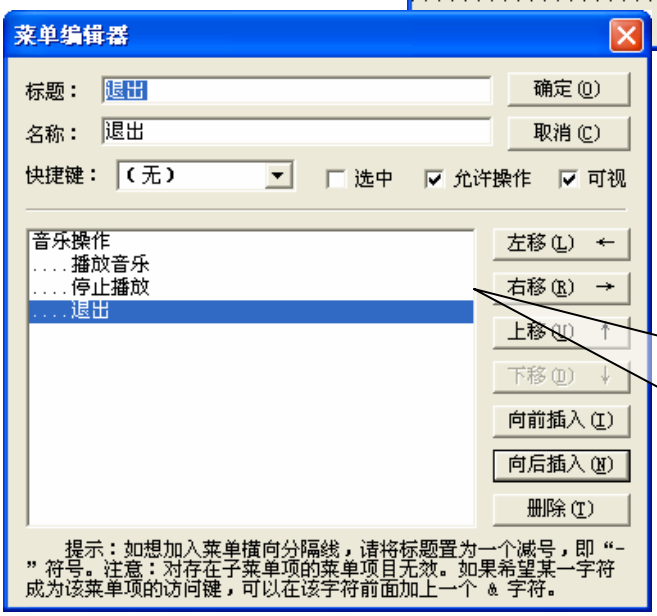
前面的菜单直接点击图片框后，从中弹出一个菜单，能不能在鼠标上方弹出呢？

本节中将使用图片资源，利用这个资源对菜单进行操作。

①下面跟着我们的一个例程来了解一下。

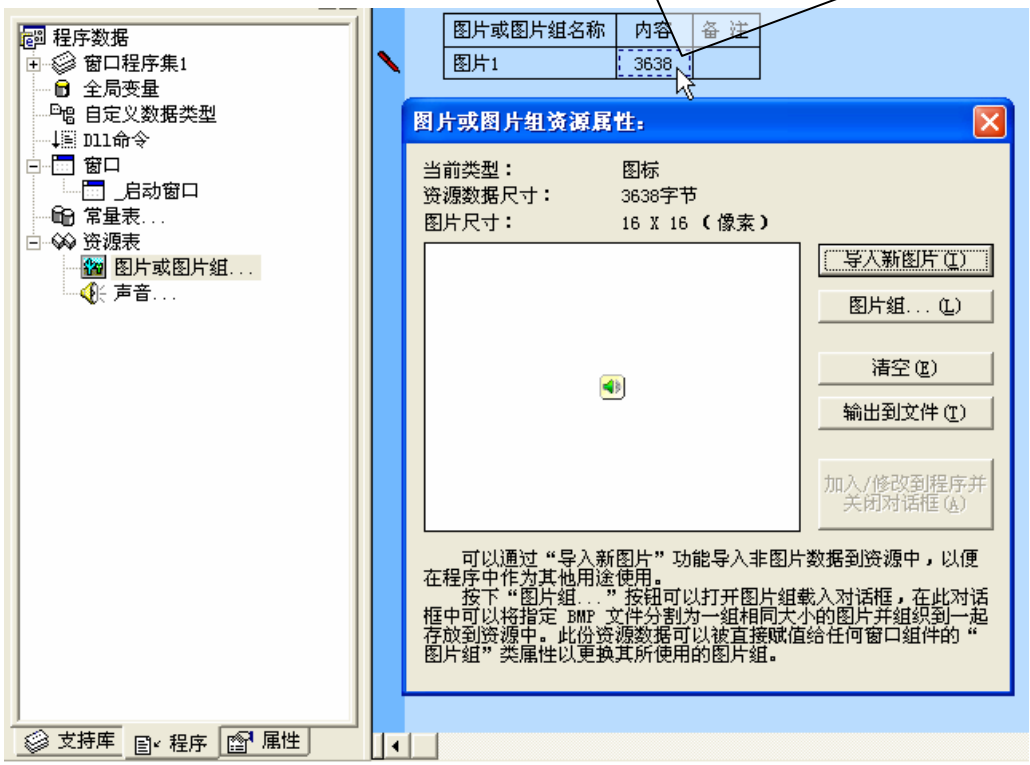
首先新建一个易程序，加入一个通用对话框控件，并加入一组菜单。





②主菜单名为：“音乐操作”，子菜单分别为：“播放音乐”、“停止播放”、“退出”。菜单属性全部为默认状态。

选择程序面板，展开其中的资源表，激活“图片或图片组”项。
③将一个图标作为图片资源导入到图片资源中。名称为：图片 1。



子程序名	返回值类型	公开	备 注
_启动窗口_创建完毕			

置托盘图标（#图片1，“请点击”）

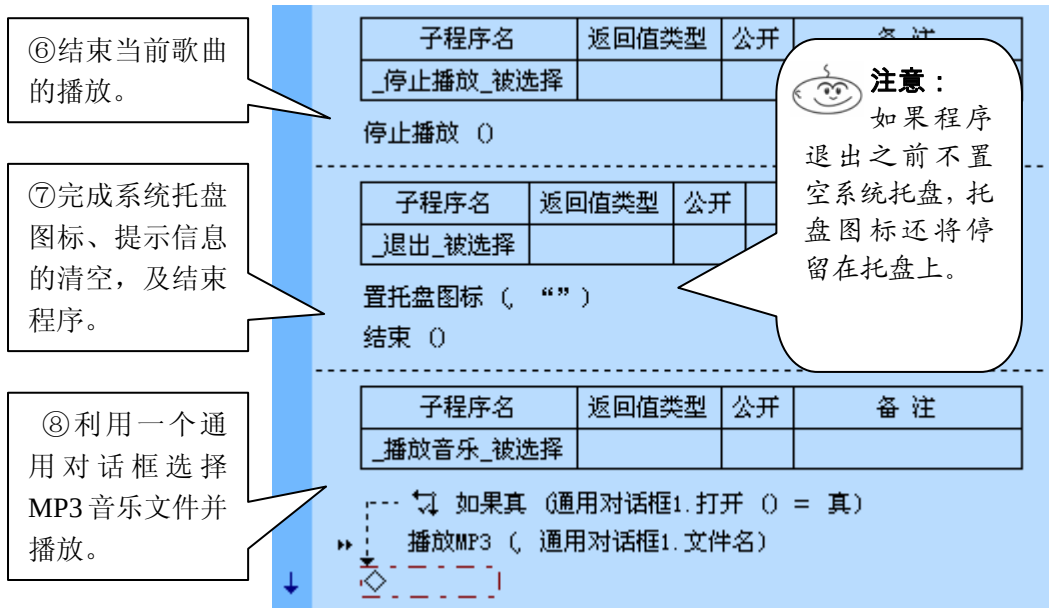
④将图片资源中的“图片 1”载入到系统托盘。

子程序名	返回值类型	公开	
_启动窗口_托盘事件			

参数名	类 型	参考	可空	数组	备 注
操作类型	整数型				

弹出菜单（音乐操作，，）

⑤用鼠标点击程序托盘，弹出“音乐操作”菜



系统托盘图标。

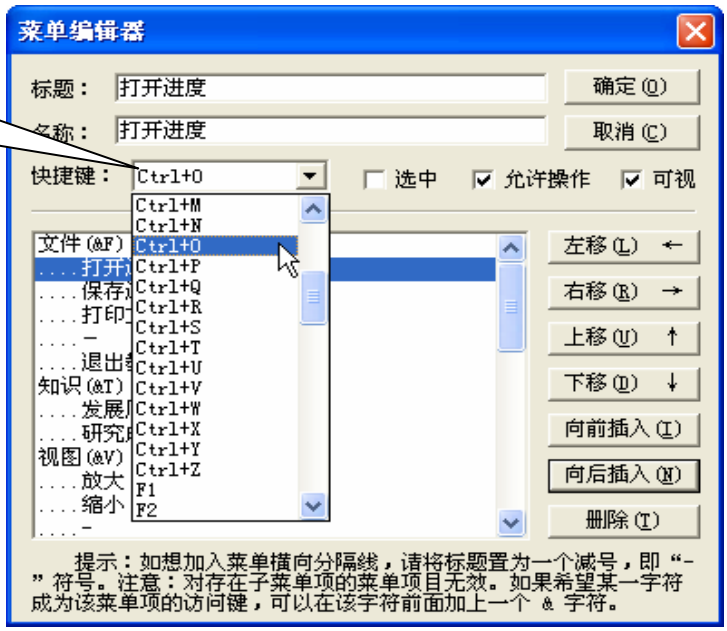
在“__启动窗口_托盘事件”事件子程序中，通过判断“操作类型”使菜单按要求弹出。如下：

```
如果真（操作类型＝3）
    弹出菜单（音乐操作，，）
```

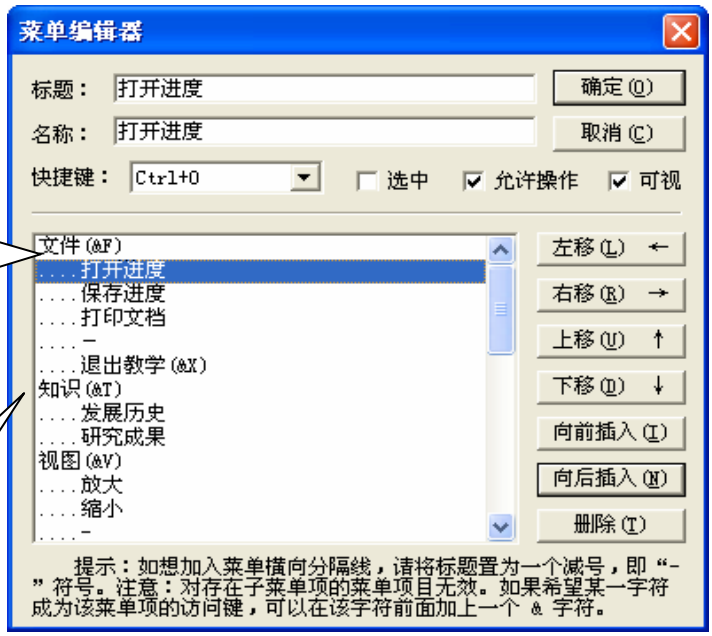
5.7 菜单中的热键和分隔符

一般程序中都设置有热键（快捷键），而它们是设置在功能相对应的菜单项上。下面把分隔符和热键的设置方法一起介绍给大家。

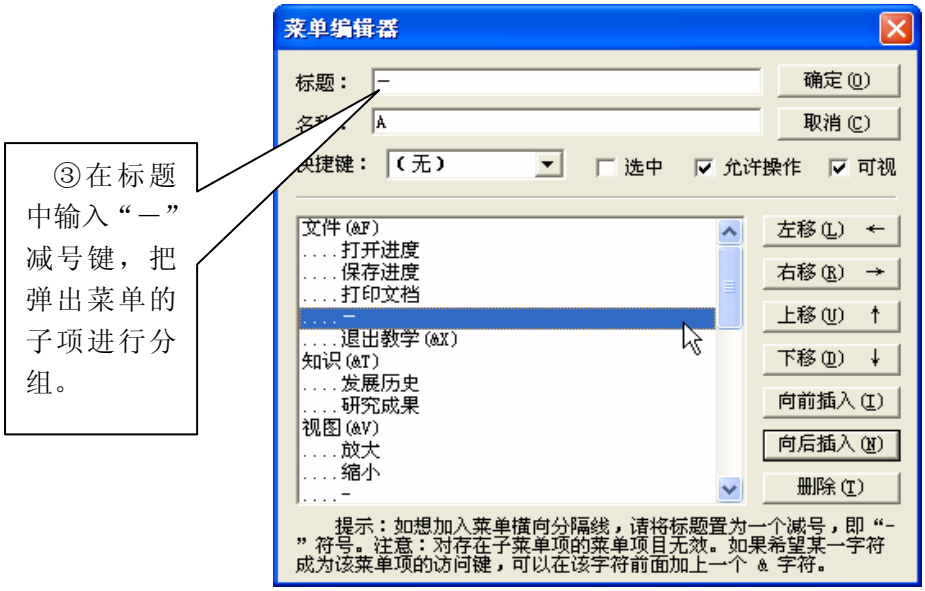
①选择“菜单编辑器”中设定的快捷键。

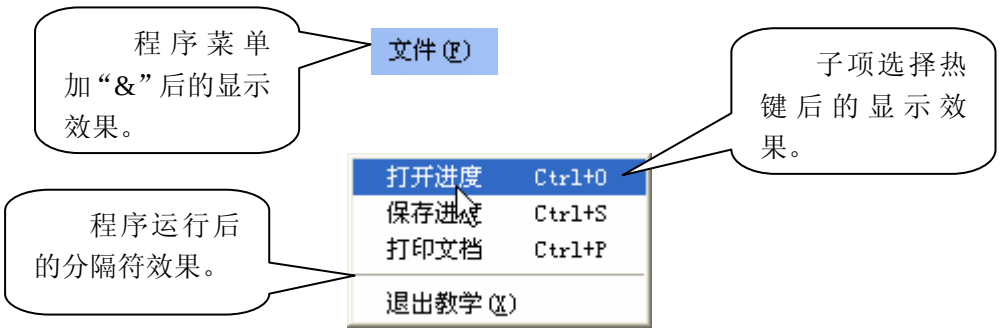


②在标题中输入“文件(&F)”把程序主菜单热键设置为“F键”。



“&”就是将其后面的一个字母加下划线





5.8 课后练习

编写两组菜单，一组为中文，一组为英文。添加一个选择框，在“_选择框_被单击”事件子程序中用“如果”判断显示其中一组菜单（用中文菜单替换英文菜单，模拟菜单被翻译）。

.

第 6 章 深入学习变量

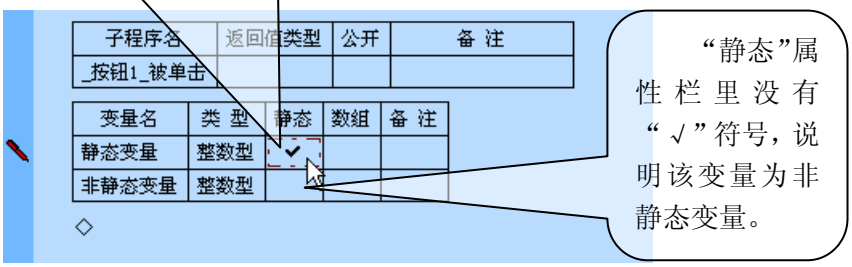
在实际编程过程中，经常用变量交换临时的数据或资源，前面已经了解了变量的概念，这一章来学习静态变量、变量数组及动态管理变量。

本章学习内容：

合理的运用变量在编程过程中，能简化或优化程序代码运行的效率。全局变量、程序集变量或局部变量，建立时默认为非静态变量，只有在子程序里局部变量才有“静态”属性设置。

6.1 认识静态变量

在子程序里面建立两个变量。分别命名为“静态变量”和“非静态变量”，用鼠标在静态变量的“静态”属性栏里点击，出现“√”符号，这样，静态变量就具有了静态属性。



1．静态变量与动态变量的区别：

非静态变量在其所在子程序开始被执行前自动分配存储空间并初始化，在所在子程序执行完毕后自动释放所分配的存储空间，也就是说，变量的存储空间仅在其所在子程序执行过程中存在；而静态变量则保留现行内容以供下次继续使用。

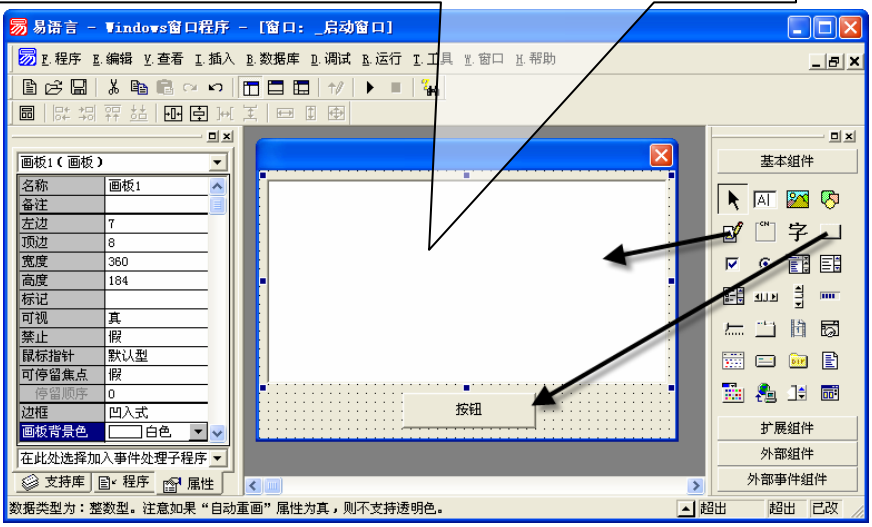
2．全局变量和程序集变量具有“静态”属性吗？

虽然在全局变量和程序集变量里没有设置“静态”选项，但是全局变量一旦赋值，变量数据就占据了指定的存储空间，并在程序运行期间永久存在仅在应用程序启动运行前被初始化一次，只有程序结束，存储空间才被释放。程序集也是一样，在程序集里面，程序集变量一旦赋值，只有程序集退出时，存储空间才被释放。

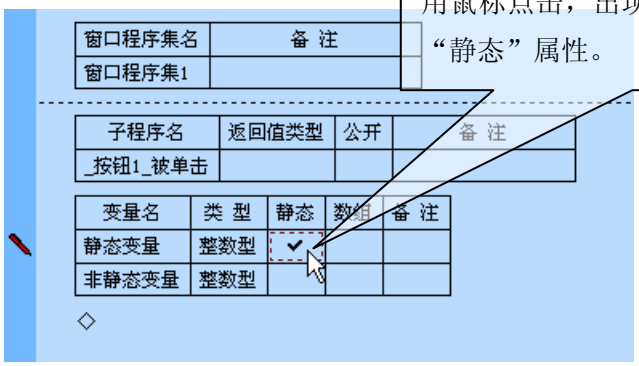
6.2 静态变量应用

下面跟着实例来演示静态变量的应用。

①新建一个易语言程序，加入一个画板和一个按钮组件，更改画板的“边框”属性为“凹入式”，“画板背景色”属性为白色。

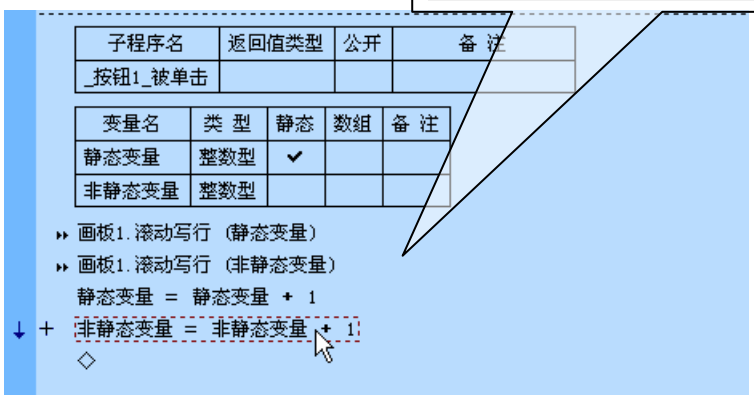


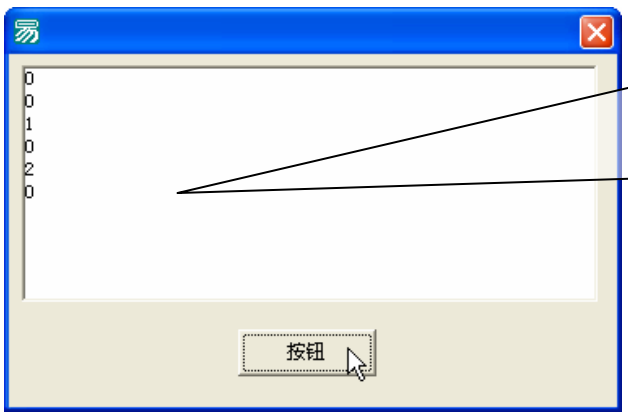
②双击按钮进入程序设计界面，在“_按钮 1_被单击”子程序里面建立两个整数型变量，分别命名为“静态变量”和“非静态变量”，在静态变量的“静态”属性栏里用鼠标点击，出现“√”符号，使其具有“静态”属性。



③顺序输入以下程序代码。

```
画板 1.滚动写行 (静态变量)
画板 1.滚动写行 (非静态变量)
静态变量 = 静态变量 + 1
非静态变量 = 非静态变量 + 1
```





④按 F5 运行程序，连续单击 3 次按钮。在画板得出如下结果。

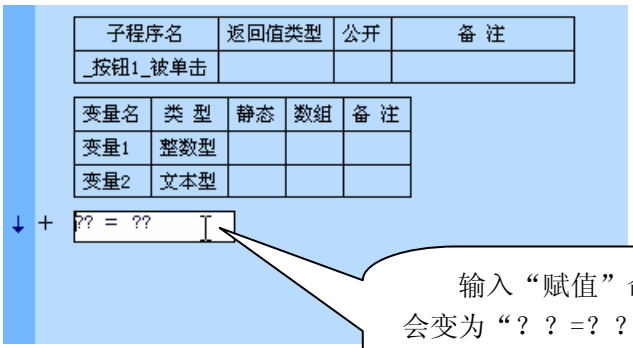
```
0
0
1
0
2
0
```

从结果可以看出，整数型静态变量和非静态变量的初始化值都是 0，子程序结束后非静态变量的数据清空，而静态变量的数据依旧保留。

6.3 变量的命令操作

变量建立好并指定了数据类型后，如果不进行赋值操作，变量会默认初始化数值。

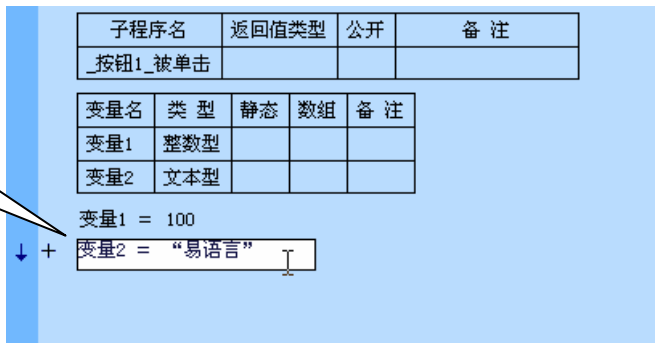
在系统核心支持库里面有变量两个操作命令。



输入“赋值”命令按回车，代码行会变为“？？=？？”，在易语言里面赋值操作的符号是“=”，后者数据赋予给前者。

可以直接输入变量名加“=”加变量值。如：

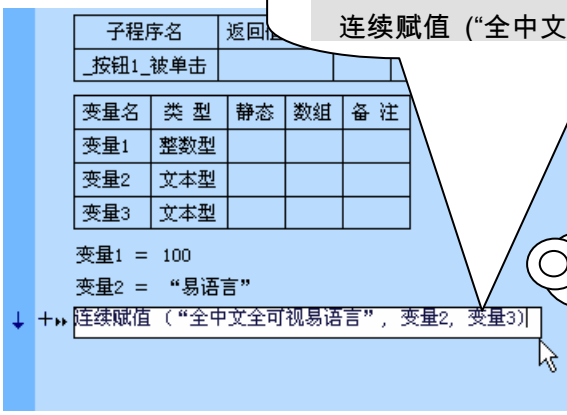
```
变量 1=100
```



加入文本型变量 3，输入“连续赋值”命令，

第一个参数是用作赋于的值或资源，第二个参数是被赋值的变量或变量数组，可以被重复添加。

连续赋值 (“全中文全可视易语言”, 变量 2, 变量 3)

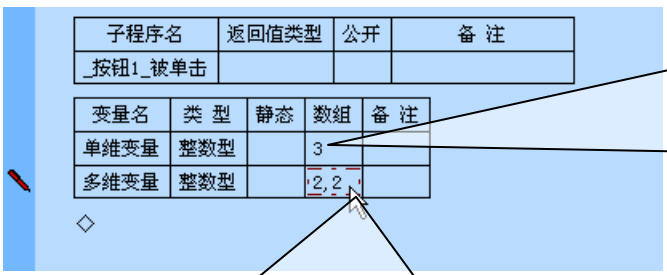


运行程序后，变量 2 和变量 3 的值都等于“全中文全可视易语言”。

这里不可以用连续赋值命令给变量 1 和变量 2 赋值，因为它们是不同类型的变量。

6.4 变量数组的定义

请打开上一章所完成的程序，将**“_按钮 1_被单击”**子程序中的内容全部删除，然后进行以下输入操作：

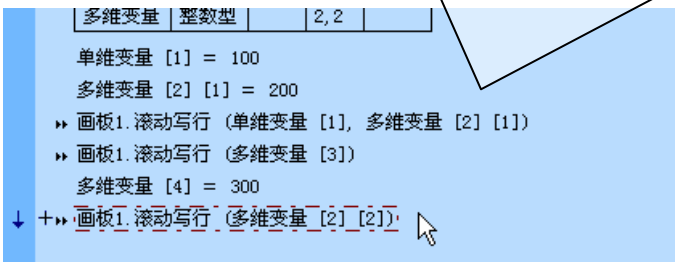


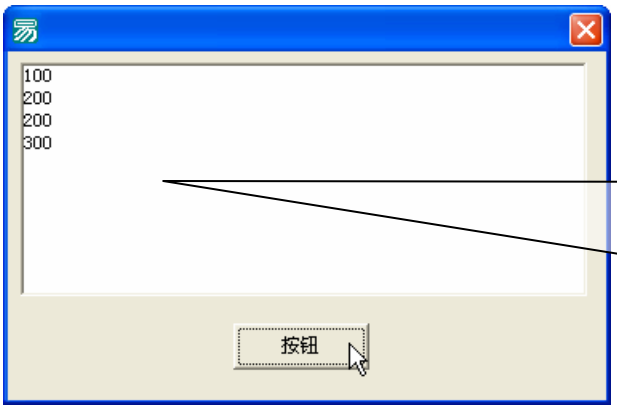
①加入一个名为单维变量的整数型变量，在其数组属性中输入“3”，表明此变量为一个单维数组，共有 3 个成员。

②加入一个名为多维变量的整数型变量，在其数组属性中输入“2，2”，表明此变量为一个二维数组，共有 4 （2×2 的结果）个成员。

③顺序输入下列语句行：

```
单维变量 [1] = 100
多维变量 [2] [1] = 200
画板1.滚动写行 (单维变量 [1], 多维变量 [2] [1])
画板1.滚动写行 (多维变量 [3])
多维变量 [4] = 300
画板1.滚动写行 (多维变量 [2] [2])
```





④按 F5 运行程序，连续单击按钮。在画板得出如下结果。

```
100
200
200
300
```

通过上面的程序可以知道：

数组变量可以有多个成员变量，每个成员变量等同于一个单独变量。各成员变量的引用方法为：

```
数组变量名 + “[” + 从 1 开始的成员位置 + “]” + .....
```

譬如上面的单维变量，它有“单维变量［1］”、“单维变量［2］”、“单维变量［3］”3 个成员。

多维变量有“多维变量［1］［1］”、“多维变量［1］［2］”、“多维变量［2］［1］”、“多维变量［2］［2］”四个成员。

该表是将经过上述运算后，各变量变更后的数值。

变量名	初始值	运行后的值
单维变量 [1]	0	100
单维变量 [2]	0	0
单维变量 [3]	0	0
多维变量 [1] [1]	0	0
多维变量 [1] [2]	0	0
多维变量 [2] [1]	0	200
多维变量 [2] [2]	0	300

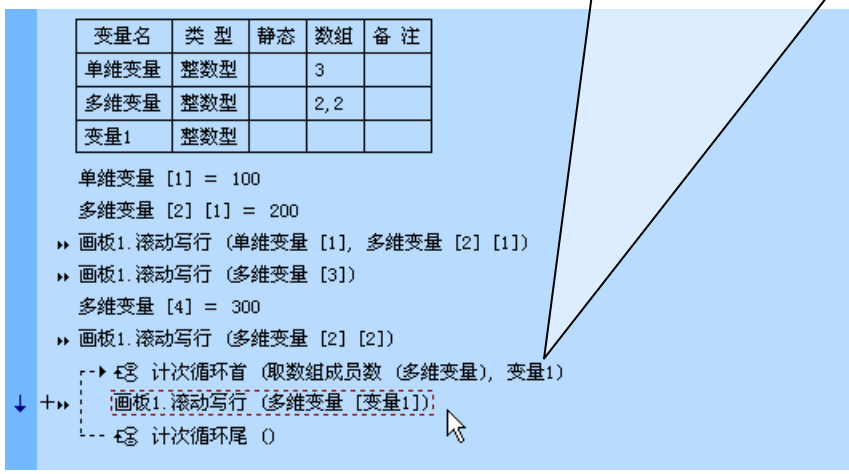
多维变量的成员也可以使用单维的方式来引用。譬如上面的“多维变量［3]”等同于“多维变量［2］[1］”，“多维变量[4］”等同于“多维变量［2］［2］”。此方法可用来遍历数组的所有成员。

请继续输入进行以下操作：

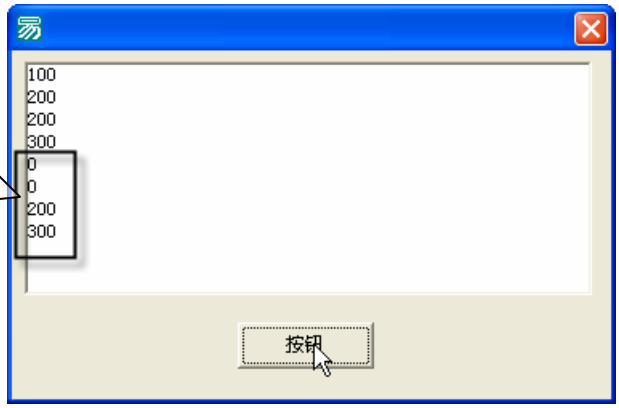
（1）加入一个名称为变量 1 的整数型变量。

（2）添加以下程序代码。

```
计次循环首 (取数组成员数 (多维变量), 变量 1)
画板 1.滚动写行 (多维变量[变量 1])
计次循环尾 ()
```



从运行结果可以看出，此段程序可以顺序显示出多维变量中所有成员的内容。其中前 4 行是第一次的运行结果，后 4 行是刚才输入新程序后的运行结果，



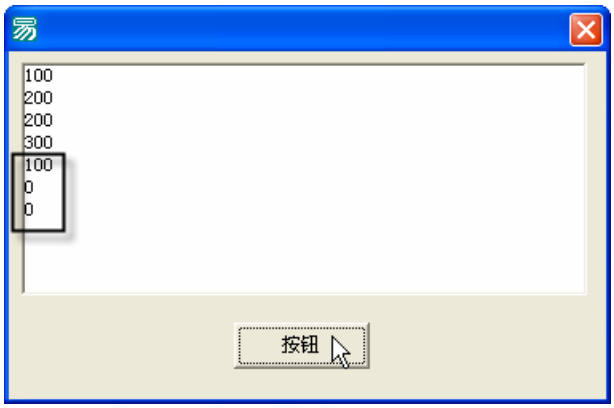
变量名	类 型	静态	数组	备 注
单维变量	整数型		3	
多维变量	整数型		2,2	
变量1	整数型			

```
单维变量 [1] = 100
多维变量 [2] [1] = 200
画板1.滚动写行 (单维变量 [1], 多维变量 [2] [1])
画板1.滚动写行 (多维变量 [3])
多维变量 [4] = 300
画板1.滚动写行 (多维变量 [2] [2])
计次循环首 (取数组成员数 (单维变量), 变量1)
画板1.滚动写行 (单维变量 [变量1])
计次循环尾 ()
```

再试一试将上面添加的程序段中的多维变量改为单维变量。

```
计次循环首 (取数组成员数 (单维变量), 变量 1)
画板 1.滚动写行 (单维变量[变量 1])
计次循环尾 ()
```

运行后查看结果。

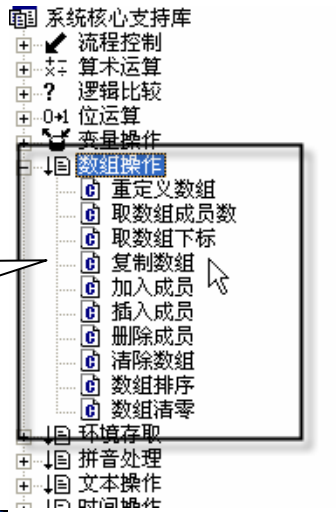


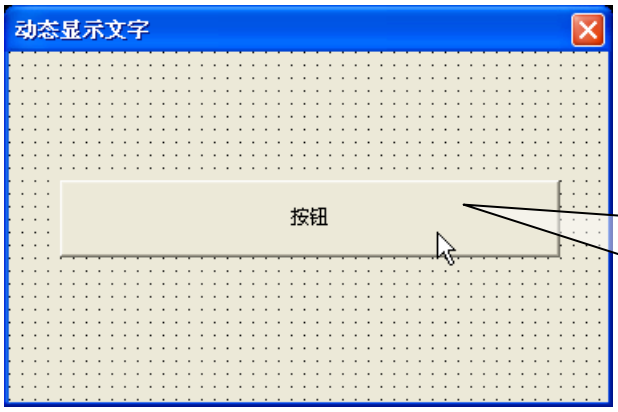
前面已说过了数组变量的每个成员可以当成单个变量应用，但是每个成员的数据类型必须相同。如果数组变量是整数型，里面对应的每个成员也必须是整数型，否则变量赋值时会提示错误。

6.5 动态管理数组变量

下面将继续讲述如何动态管理数组变量，它是易语言的高级特性之一。

在系统核心支持库下有一组数组操作命令，我们通过一个简单的例子来了解它们。





①在窗体上创建一个按钮组件。把其宽度拉长，更改窗口标题为“动态显示文字”。

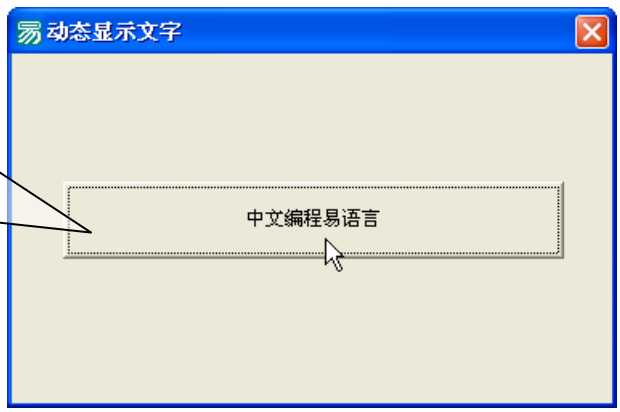
②分别建立文本变量 1 和整数型变量 2，在变量 1 的数组属性里输入 0，请继续输入进行代码：

```
重定义数组 (变量1, 假, 7)          //把变量1的成员数重定义为7
变量1={ “中”, “文”, “编”, “程”, “易”, “语”, “言” }
                                   //赋值给变量1每个成员对应的文本
按钮1.标题=“”                      //清除按钮1的标题文字
计次循环首 (取数组成员数 (变量1), 变量2)  //循环次数为变量1的成员数
    按钮1.标题 = 按钮1.标题 + 到文本 (变量1 [变量2])
                                   // 取变量1每个成员文字给按钮1的标题
    延时（300）                     //每次取变量1成员内容间隔的时间
计次循环尾 ()
```

变量名	类 型	静态	数组	备 注
变量1	文本型		0	
变量2	整数型			

```
重定义数组 (变量1, 假, 7)
    ※欲重定义的数组变量： 变量1
    ※是否保留以前的内容： 假
    ※数组对应维的上限值： 7
变量1 = { “中”, “文”, “编”, “程”, “易”, “语”, “言” }
按钮1.标题 = “”
计次循环首 (取数组成员数 (变量1), 变量2)
    按钮1.标题 = 按钮1.标题 + 到文本 (变量1 [变量2])
    延时 (300)
计次循环尾 ()
```

按 F5 运行程序，点击按钮 1，按钮的标题会动态显示“中-文-编-程-易-语-言”。



下面删除代码段：

```
重定义数组 (变量 1, 假, 7)
```

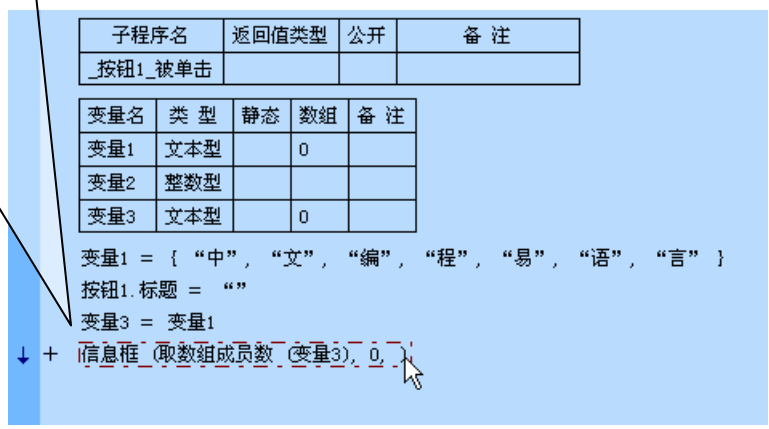
再运行程序，发现结果一样，所以得出结论：

```
变量 1 = { “中”, “文”, “编”, “程”, “易”, “语”, “言” }
```

这种赋值给变量 1 的操作，可以重新改变变量成员数目，并清除原先所有数据。原来成员数为 0 个成员，执行后改为 7 个成员。

③再清除循环的一组代码，添加一个新的文本型变量 3，数组成员为 0，继续输入：

```
变量 3=变量 1
```



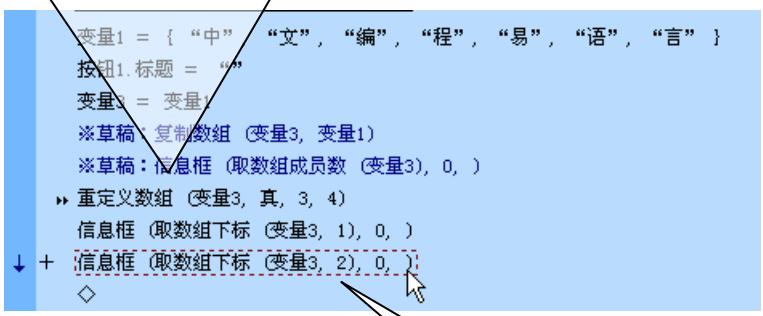
④运行后，弹出信息框，得出“7”个变量 3 的成员数，也就是说明：**变量 3=变量 1**

这行代码把变量 1 的成员数及其内容同时赋给变量 3，等同于：“复制数组”命令：

复制数组 (变量 1, 变量 3)

⑤用 Ctrl+回车键把信息框代码行置为草稿，接着输入以下代码：

```
重定义数组 (变量 3, 真, 3, 4)                    //把单维变量 3 重定义为二维变量

信息框 (取数组下标 (变量 3, 1), 0, )    //得出变量 3 第一维的成员数
```



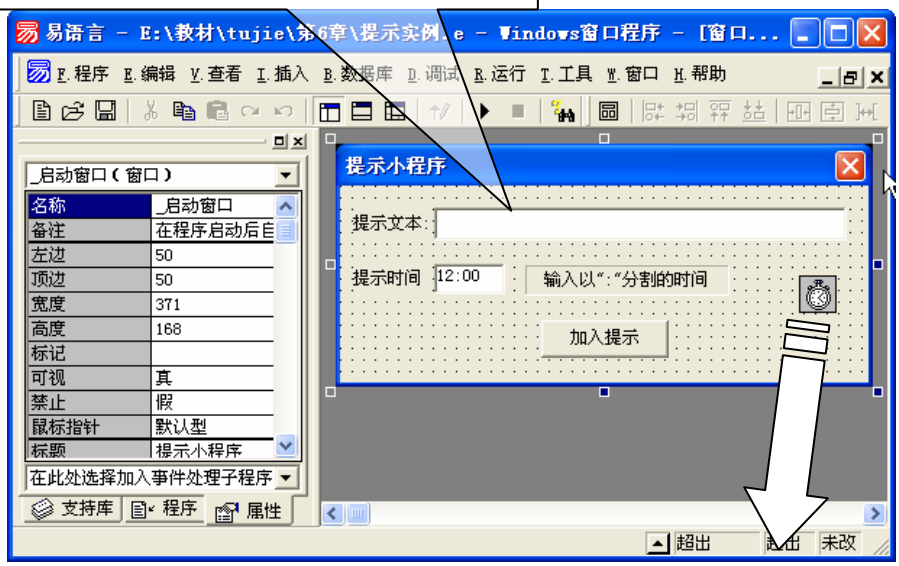
运行后得出变量 3 第一维的成员数为 3；第二维的成员数为 4。

6.6 定时提醒小程序练习

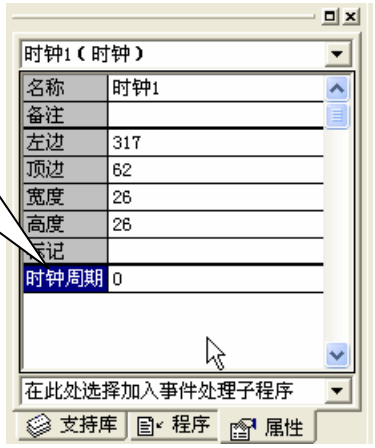
变量数组是编程中临时存储多个数据的“小仓库”，其类型可以是整数、文本、字节集等等，在实际编程中会常常读成员内容时，遇到超出下标的错误，好多出现在循环代码中，需注意！

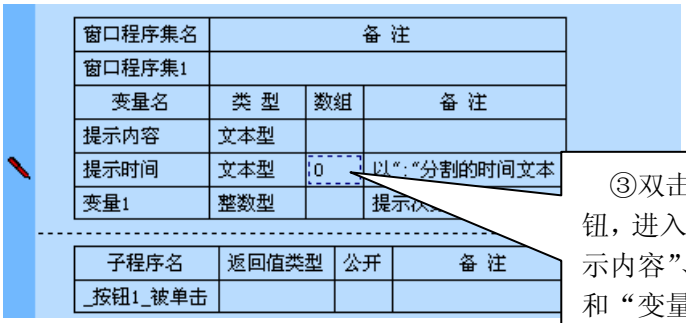
下面就来练习本节的定时提醒小程序。

①在新建的易程序窗口上添加如图所示的组件，调整位置大小并更改标题内容。



②选住时钟组件，在属性面板上更改“时钟周期”为 1000，单位是毫秒，1 秒=1000 毫秒，目的是每 1 秒都执行对应的周期事件。

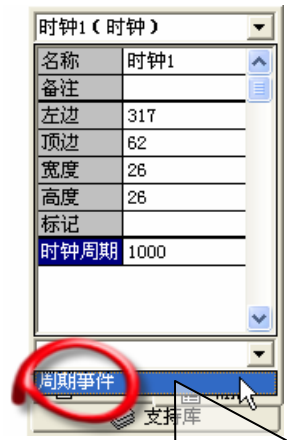


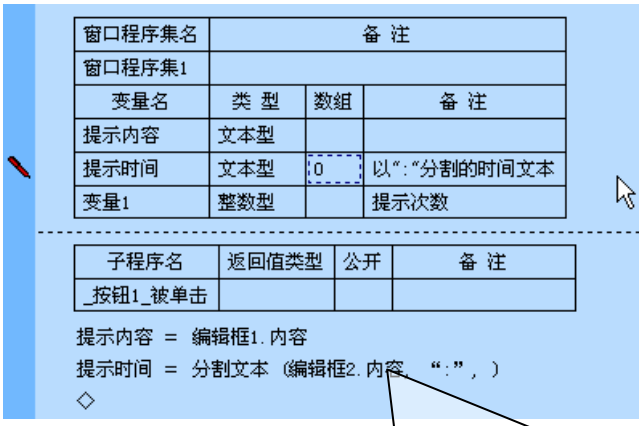


窗口程序集名	备 注		
窗口程序集1			
变量名	类 型	数组	备 注
提示内容	文本型		
提示时间	文本型	0	以":"分割的时间文本
变量1	整数型		提示...

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

③双击标题为“加入提示”的按钮，进入代码编写窗口。新建 “提示内容”、“提示时间”两个文本型和“变量 1”整数型程序集变量，并设置“提示时间”变量的数组属性为 0 个成员。

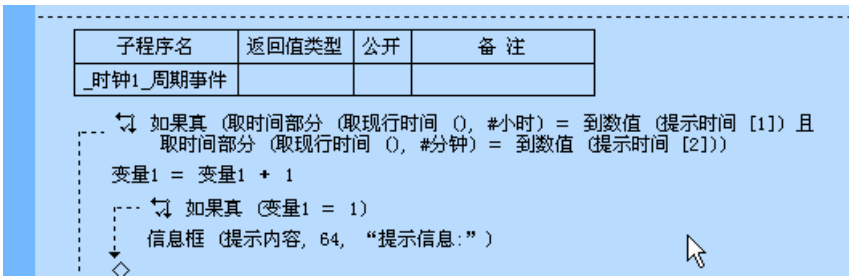




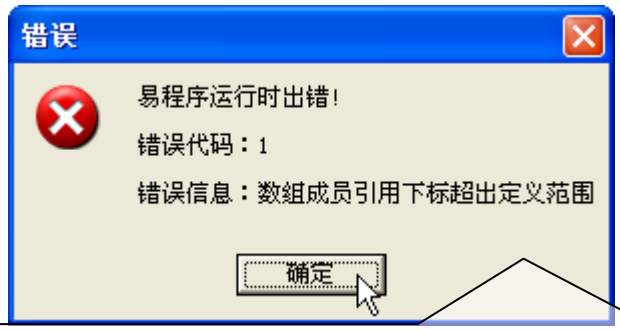
④在“_按钮 1_被单击” 子程序下输入：

```
提示内容 = 编辑框 1.内容
提示时间 = 分割文本 (编辑框 2.内容, ":", )
```

⑤切换到时钟 1 属性面板，点击独有的周期事件，进入“_时钟 1_周期事件”子程序代码编写窗口，也可以直接用双击时钟 1 组件进入“_时钟 1_周期事件”子程序代码编写窗口。输入以下代码：



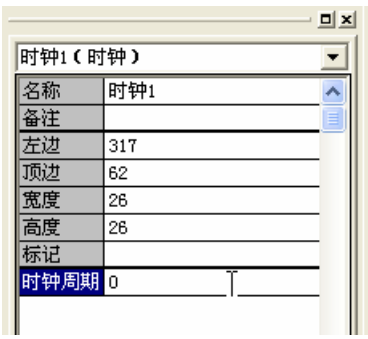
上面的命令代码是取现在时间的小时部分和分钟部分与指定提示的时间判断，相同则执行信息框提示。变量 1 的目的是控制一分钟只提示一次，当然你也可以让程序提示两次或多次。只需更改“如果真 (变量 1 % 10 = 0)”代码，等于每间隔 10 秒 提示一次。

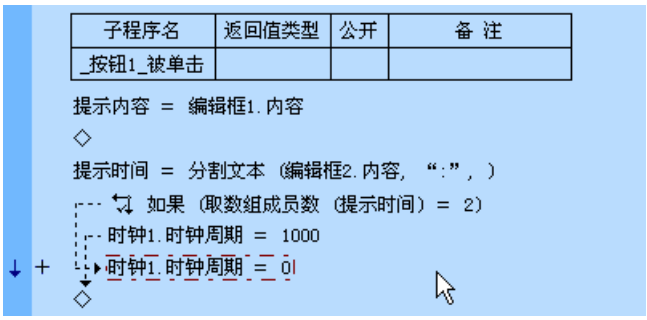


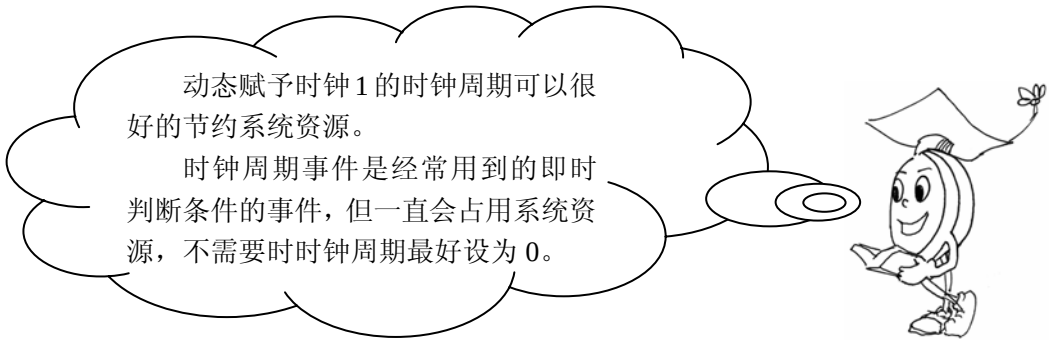
⑥按钮 F5 运行程序，立刻会弹出错误信息，问题在哪里呢？原来“提示时间”数组变量的成员默认是 0，判断提示时间[1]当然出错了。

先把时钟 1 的时钟周期设为 0，再“_按钮 1.被单击”子程序中添加代码，判断分割后的提示时间数组的成员数，如果等于 2，就分配时钟 1 的时钟周期为 1000，不等于 2 还设为 0。

```
如果 (取数组成员数 (提示时间) = 2)
    时钟 1.时钟周期 = 1000
时钟 1.时钟周期 = 0
```







⑦按钮 F5 运行程序，在提示文本后面编辑框输入一段文字，“我爱易语言！”，输入提示时间，格式为“小时:分钟”，因为程序是分割“:”到提示时间数组，点击“加入提示”按钮，程序到指定时间会谈出提示的内容。

上面提示程序的原理就是输入时间数字，以“:”分割出小时和分钟与实际时间判断，相符则执行提示信息。如果输入带“：”符号的数字是分解不成功的。可以在_按钮 1_被单击子程序里加入如下代码：

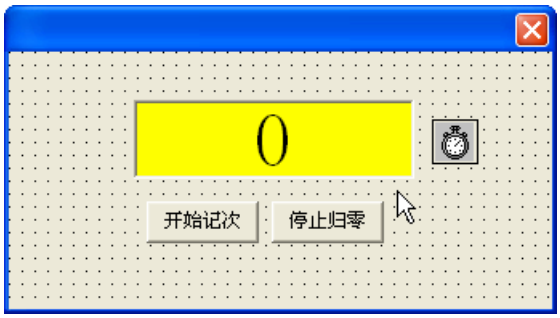
```
如果真 (取数组成员数 (提示时间) ≠ 2)
    提示时间 = 分割文本 (编辑框 2.内容, “:”, )
```

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

```
提示内容 = 编辑框1.内容
◇
提示时间 = 分割文本 (编辑框2.内容, “:”, )
如果真 (取数组成员数 (提示时间) ≠ 2)
    提示时间 = 分割文本 (编辑框2.内容, “：”, )
◇
如果 (取数组成员数 (提示时间) = 2)
    时钟1.时钟周期 = 1000
    时钟1.时钟周期 = 0
◇
```

6.9 课后练习

（1）说一说静态变量与动态变量的区别，并利用时钟组件制作一个秒表记次程序。



窗口程序集名	备 注		
窗口程序集1			
变量名	类 型	数组	备 注
停止	逻辑型		

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

```
如果 (时钟1.时钟周期 = 0)
    时钟1.时钟周期 = 1000
    按钮1.标题 = “暂停”
    时钟1.时钟周期 = 0
    按钮1.标题 = “开始计次”
```

子程序名	返回值类型	公开	备 注
_时钟1_周期事件			

变量名	类 型	静态	数组	备 注
记次变量	整数型	✔		

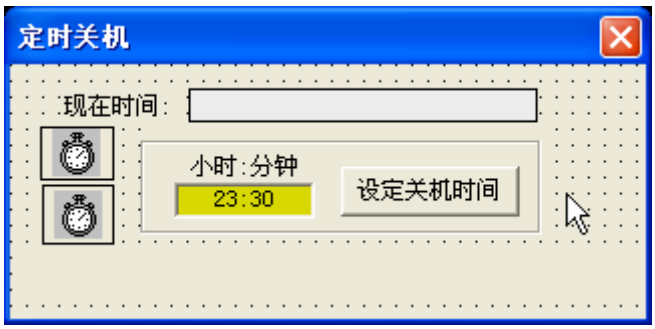
```
如果真 (停止 = 真)
    记次变量 = 0
    停止 = 假
记次变量 = 记次变量 + 1
标签1.标题 = 到文本 (记次变量)
```

子程序名	返回值类型	公开	备 注
_按钮2_被单击			

```
标签1.标题 = “0”
时钟1.时钟周期 = 0
按钮1.标题 = “开始计次”
停止 = 真
```

（2）练习把单维数组变量的内容改为多维数组变量，并保留变量内以前的数据。

（3）根据定时提醒小程序的原理，练习制作一个定时关机程序。



窗口程序集名	备 注		
窗口程序集1			
变量名	类 型	数组	备 注
关机时间	文本型	2	

子程序名	返回值类型	公开	备 注
_时钟1_周期事件			

```
标签3.标题 = “您的关机时间为：” + 关机时间 [1] + “点” + 关机时间
    [2] + “分钟”
处理事件 ()
    如果真 (取时间部分 (取现行时间 (), 6) = 到数值 (关机时间 [1]) 且
        取时间部分 (取现行时间 (), 7) ≥ 到数值 (关机时间 [2]))
    关闭系统 (1, 真)
```

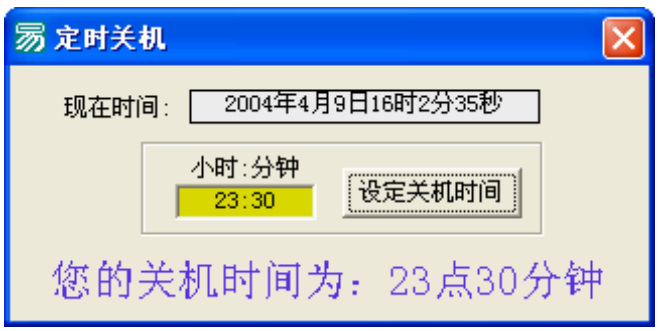
子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
临时	文本型		2	

```
如果真 (寻找文本 (编辑框2.内容, “:”, , 假) = -1 且 寻找文本 (编辑框2.内容, “：”, , 假) = -1)
    信息框 (“输入时间格式不对!，请输入类似18:18格式的时间”, 16, )
    标签3.标题 = “您的关机时间为：” + “没有设置”
    时钟1.时钟周期 = 0
    返回 ()
临时 = 分割文本 (编辑框2.内容, “:”, )
如果真 (临时 [1] = “”)
    临时 = 分割文本 (编辑框2.内容, “：”, )
关机时间 = 临时
时钟1.时钟周期 = 1000
```

子程序名	返回值类型	公开	备 注
_时钟2_周期事件			

```
编辑框1.内容 = 到文本 (取现行时间 ())
```



第七章 组件的使用

本章主要介绍“易语言”中的组件是如何添加和使用的，并用几个简单的小例程来了解组件的属性，事件，和方法。

本章学习内容：

相信大家已经使用过易语言中的好多种组件了，对组件已经不陌生了，但你知道吗，组件中的很多属性都有它特殊的用途，并且，有很多组件都有它特有的事件和方法，只有对这些事件和方法都了解，编程时才更加得心应手了。

7.1 如何用组件设计程序界面

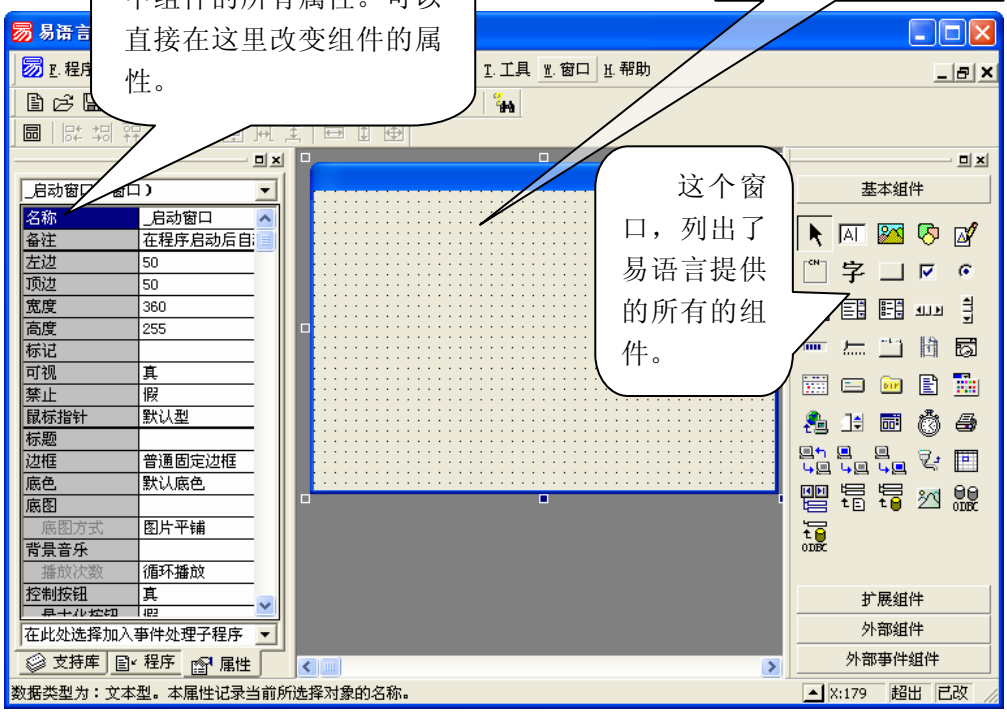
很多软件都有漂亮的可视化界面，那么这些漂亮的软件界面是如何创建的呢？这就是组件的作用了，本节就来教大家如何为自己的程序设计一个漂亮的界面。

要想对程序的界面进行设计，我们就要先了解组件的各种不同的属性。

首先，新建一个程序，产生一个窗口。

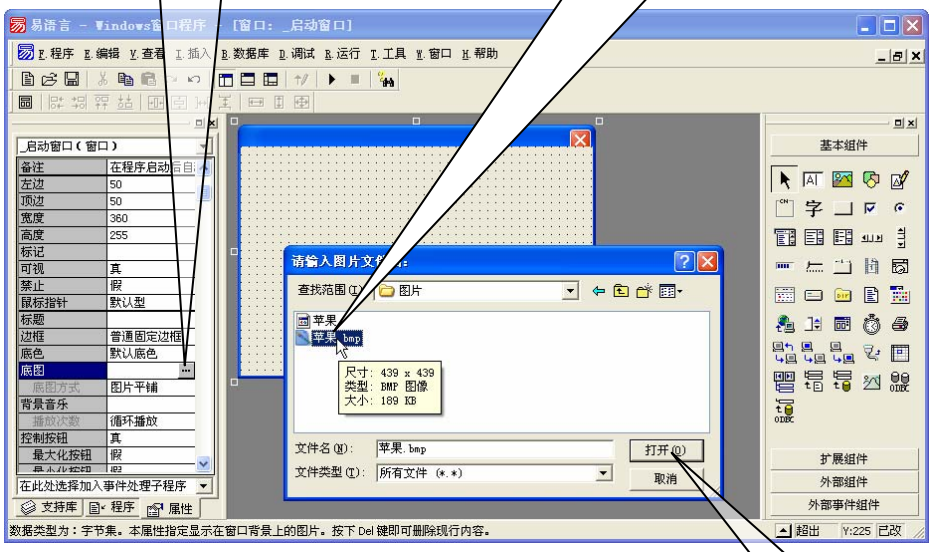
这个窗口显示了被选中组件的所有属性。可以直接在这里改变组件的属性。

这个窗口，列出了易语言提供的所有的组件。



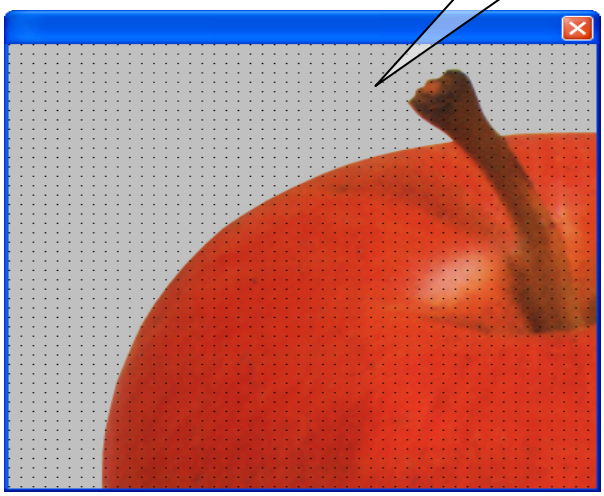
①点击底图属性旁边的小按钮。

②在弹出的窗口中选择一张精美的图片。

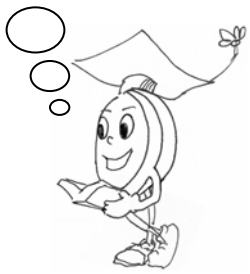


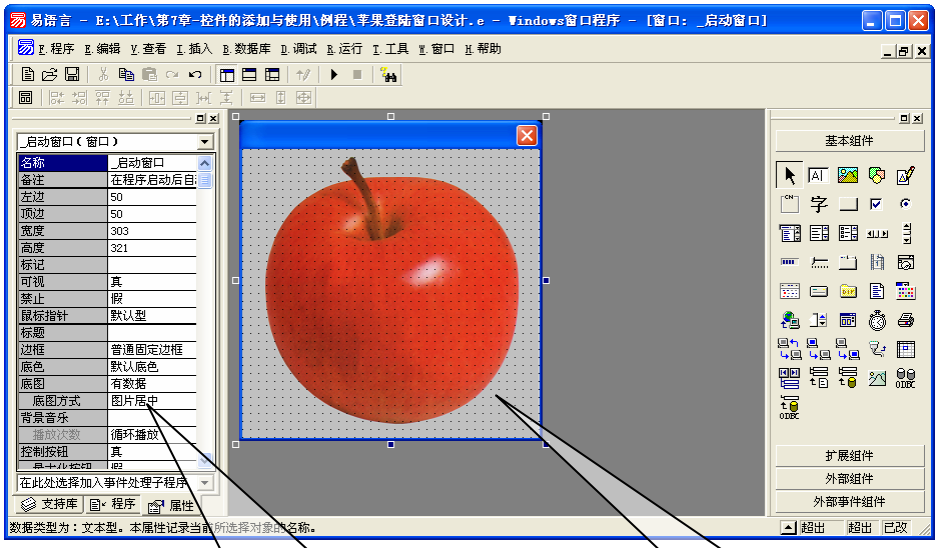
刚加完底图的窗口。

③然后点击打开按钮。



哇！出现了这么大的一个苹果图片，怎么办呢？





④如果底图过大，可以给底图方式属性设置成“图片居中”

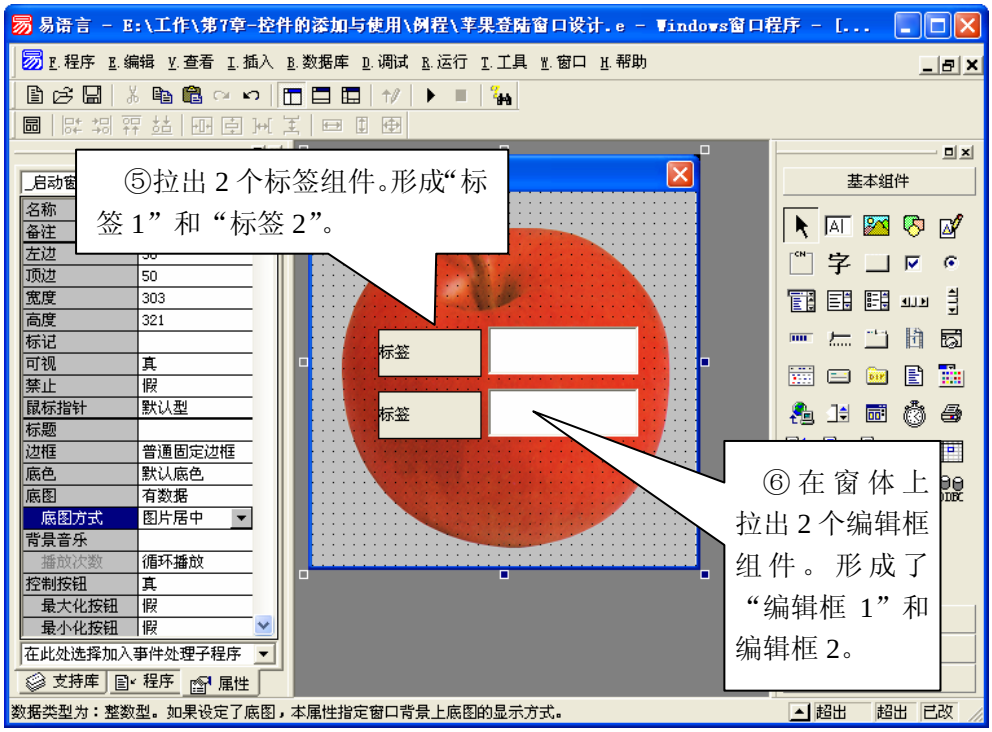
刚才的大图片已经调整成合适的大小了。

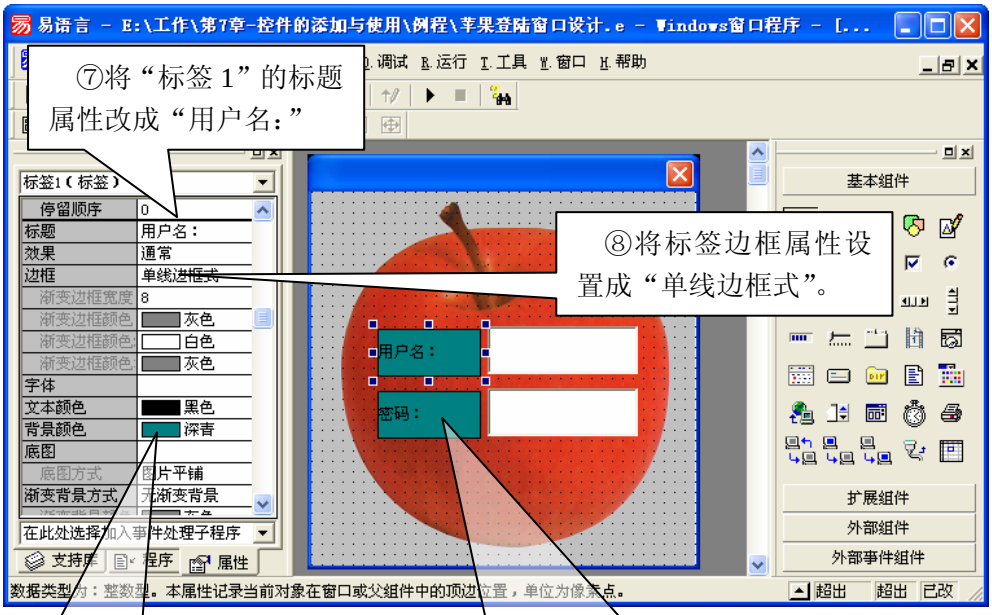
注意：如果添加了图片后又想将图片删除怎么办？

可以选中底图属性，然后点击鼠标右键，在弹出菜单中选择删除内容即可。

可者选中“底图”属性后，按[Del]键删除。

仅仅给窗体了一张图片，就得到了一个与众不同的苹果窗口，我们继续将这个苹果窗口设计成一个软件的用户登陆窗口吧！



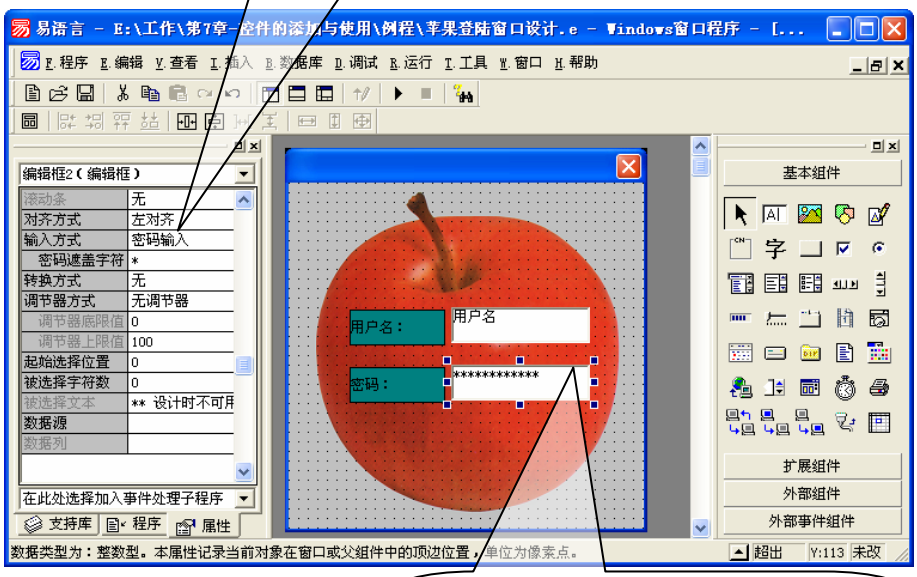


⑨将标签 1 的背景颜色属性设置成深青色。

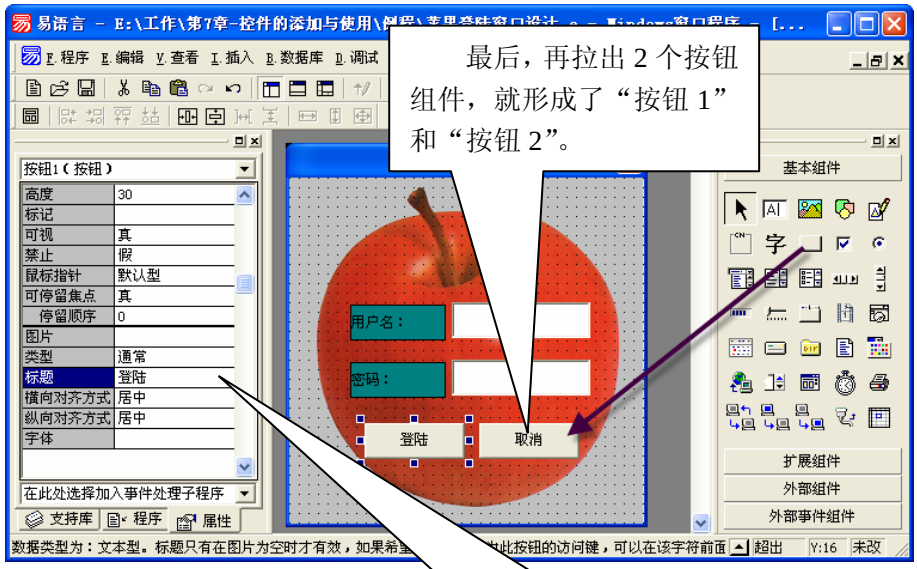
用修改标签 1 的方法，来修改标签 2 的属性。不同的是将标签 2 的标题改成“密码：”

都用过自动提款机吧？输入密码的地方是不能显示出来了，我们设计的登陆窗体中的“编辑框 2”也是用来输入密码的，所以也要作好保密工作呀！

⑩将编辑框 2，即输入密码的编辑框的输入方式属性设置成“密码输入”。



运行以后，由于编辑框 2 的输入方式改成“密码输入”，输入的内容就保密了，输入的密码都用星号显示。



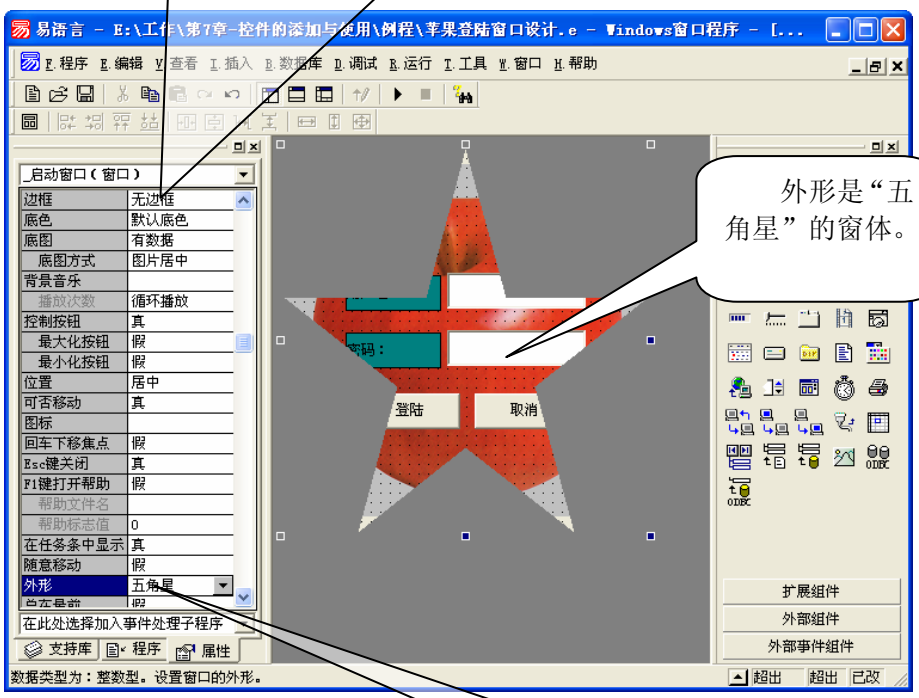
将 2 个按钮的标题属性，分别改成“登陆”和“退出”。

好拉，一个登陆窗口就做好了，怎么样，好看吗？这个窗口可以用来控制一个软件的使用，只有输入正确的用户名和密码以后，才可以继续使用软件，这样其他人就不能乱用你编写的程序了。

当然，光有窗口是不行的，还要在按钮组件被单击的事件中输入其他的代码才行。组件的事件如何使用呢？下面将在介绍控件事件的时候做详细介绍。

你还想得到更加特别，更加与众不同的窗口吗？易语言可以帮助你实现。易语言为你准备了 29 种精美而特别的窗体外形。

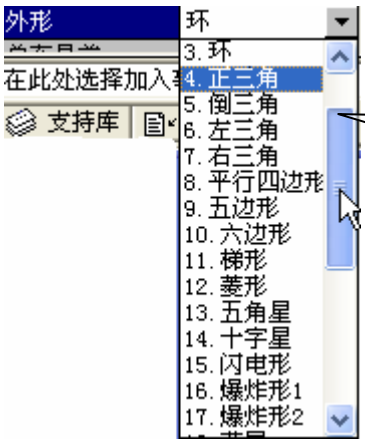
改变窗体外型：①将窗体组件的边框属性设置成“无边框”的



外形是“五角星”的窗体。

②将窗体的外形属性设置成“五角星”。

设计完更加特别的窗体外形以后，你的窗体就焕然一新了。还可以换上其他的外形看看，哪个更漂亮？



外形属性的的小箭头被点击后，就列出了29中外型的被选项。可以给窗体更换不同的外型。

7.2 动态修改组件属性

修改了这么多组件的属性，都是手动的去修改，这些修改工作是要在编写程序以前就做好的。

那在编写程序的时候想要随时的用代码修改组件的属性可以吗？

其实组件的属性都可以用代码的形式在程序运行的过程中动态的去改变。动态改变组件属性的方法也是最常用和最灵活的方法。下面就介绍一下如何动态的来修改组件的属性。

动态改变组件属性的基本方法是：

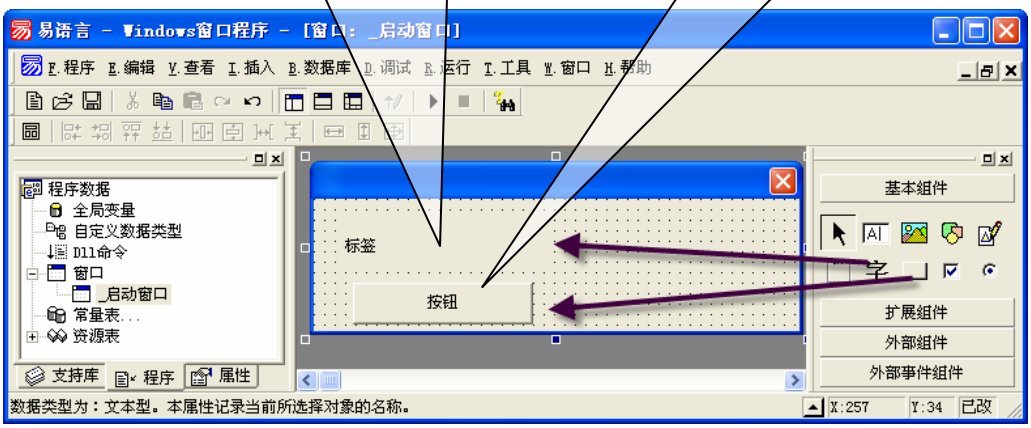
```
组件名.组件属性=要给组件属性的值
```

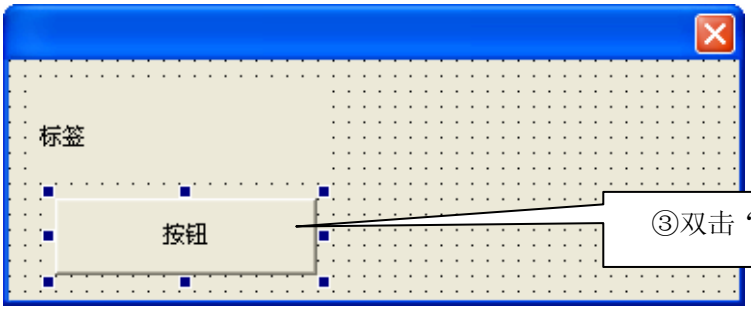
把刚才编写的登陆窗口保存好，记住保存在哪，下面会使用到它。

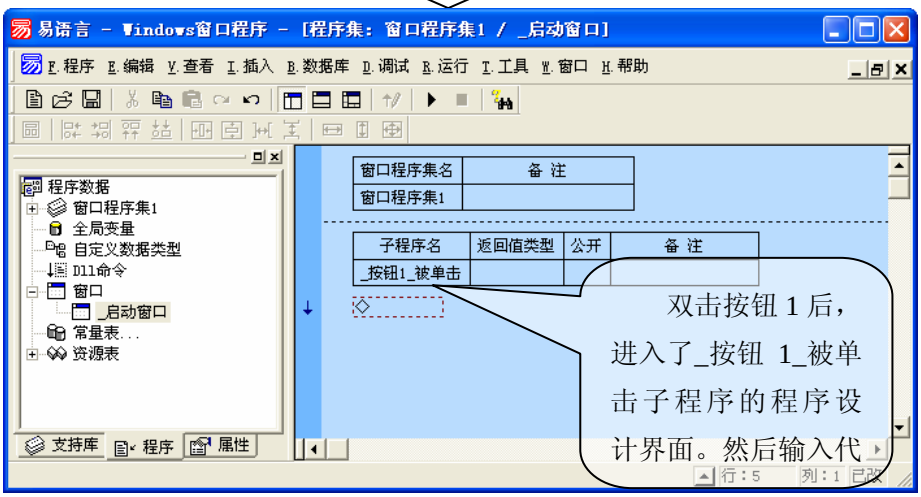
下面跟着例子新建一个易程序。

①点击标签组件后，在窗体中拉出一个标签组件。形成“标签 1”。

②点击按钮组件后，在窗体中拉出一个按钮组件。形成“按钮 1”。



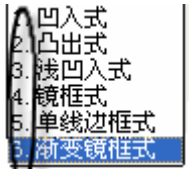




动态的改变标签的标题属性。

让标签的位置发生改变，代码运行时反复调用该属性的前一个值。

改变标签的边框。



程序名	返回值类型	公开	备 注
_按钮1_被单击			

```
标签1.标题 = “动态改变标签属性”
标签1.左边 = 标签1.左边 + 10
标签1.顶边 = 标签1.顶边 + 1
标签1.边框 = 6
标签1.背景颜色 = 取随机数 (255000, 255255000)
```

随机改变标签的背景颜色。由于颜色属性其实是一个数值型的，所以用

```
取随机数 (255000, 255255000)
```

注意：在给属性动态改变属性的时候，一定要注意属性本身的数据类型。也就是说，文本型的属性就给这个属性一个文本型的数据，整数型的属性就给这个属性一个整数型的数据。

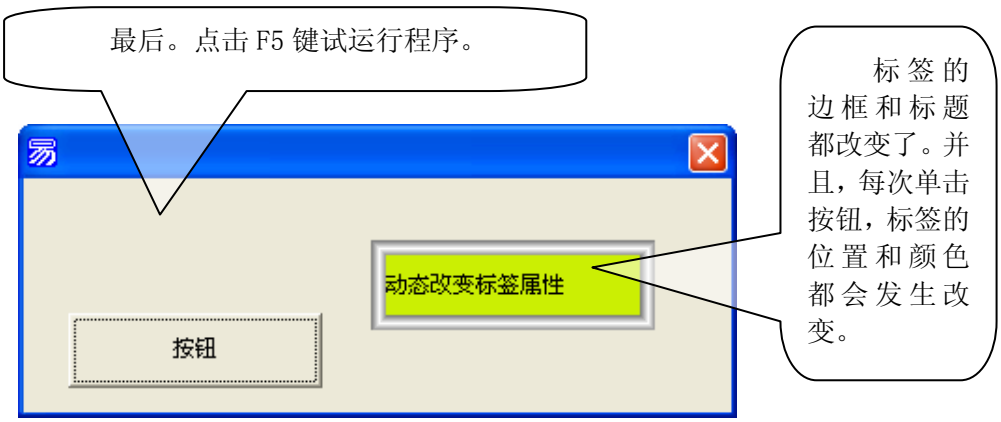
比如：

```
标签1.标题 = “动态改变标签属性”
```

标签的标题属性是文本型的，所以改变它的时候就要把给它的值用双引号括起来，这样就表示了一段文本型的数据。即代码中的“动态改变标签属性”。

```
标签1.边框 = 6
```

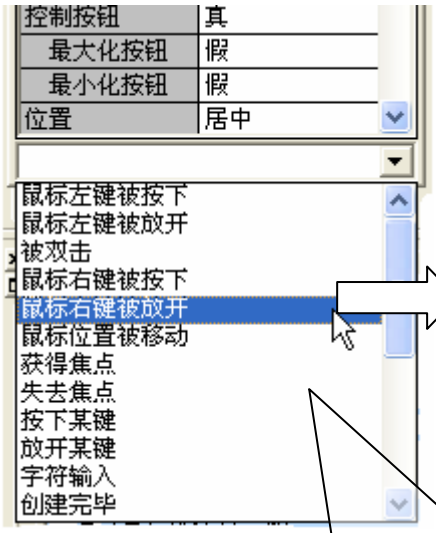
标签的边框属性是整数型的，所以改变边框属性时就要给边框属性一个整数型的数据，即代码中的6。



7.3 初步了解组件事件

每个组件都有它的事件，组件的事件就是当在一个组件上发生了某种事件后，就会运行这个事件相应的子程序。比如，_启动窗口的鼠标右键被按下的事件，选中后就会产生“_启动窗口_鼠标右键被按下”的子程序，我们在下面输入了代码，当程序运行以后，在启动窗口上点击鼠标右键，就会运行“_启动窗口_鼠标右键被按下”的子程序，同时，就执行了在这个子程序下输入的全部代码。

有些事件是每个组件都有的，叫做基本事件，还有些事件是某些组件所独有的，叫这个组件的自有事件。



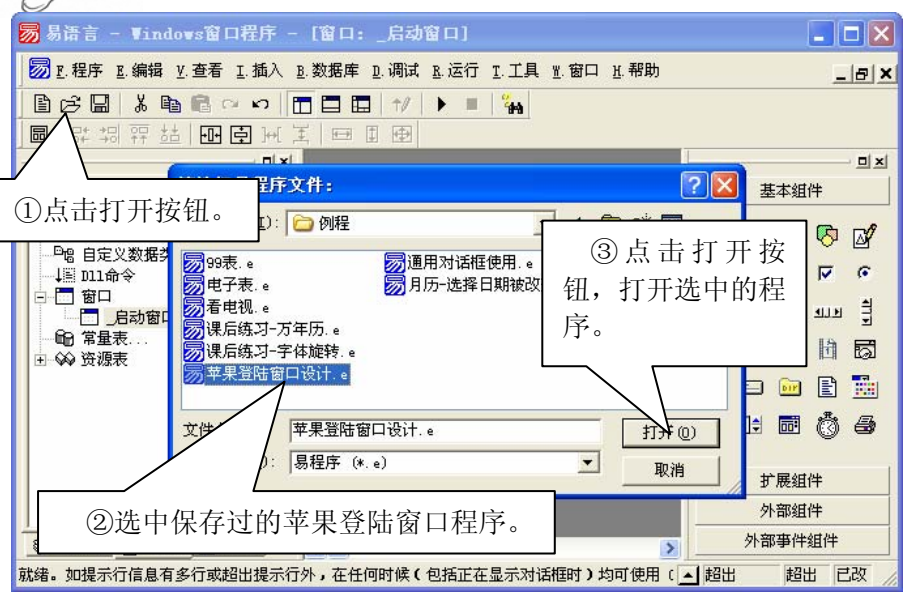
在一个组件的事件菜单中选择了一个事件以后，就会产生这个组件发生该事件的子程序。

子程序名	返回值类型	公开	备 注		
__启动窗口_鼠标右键被放开	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

每个组件属性窗口的最下方，都有一个列出这个组件所有事件的菜单。

如果发生了在_启动窗口中右键被按下的事件以后，就会执行这个事件下编写的代码。

还记得前面编写的苹果图案的登陆窗口吗？找到并打开它。

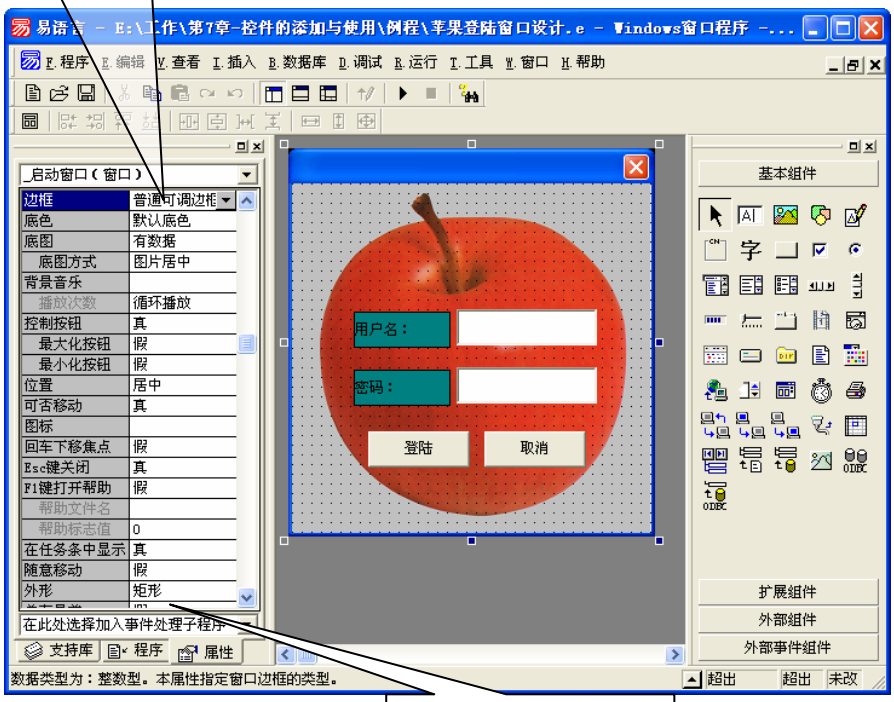


①点击打开按钮。

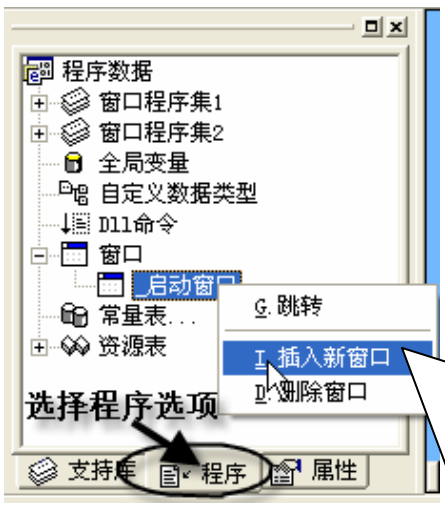
②选中保存过的苹果登陆窗口程序。

③点击打开按钮，打开选中的程序。

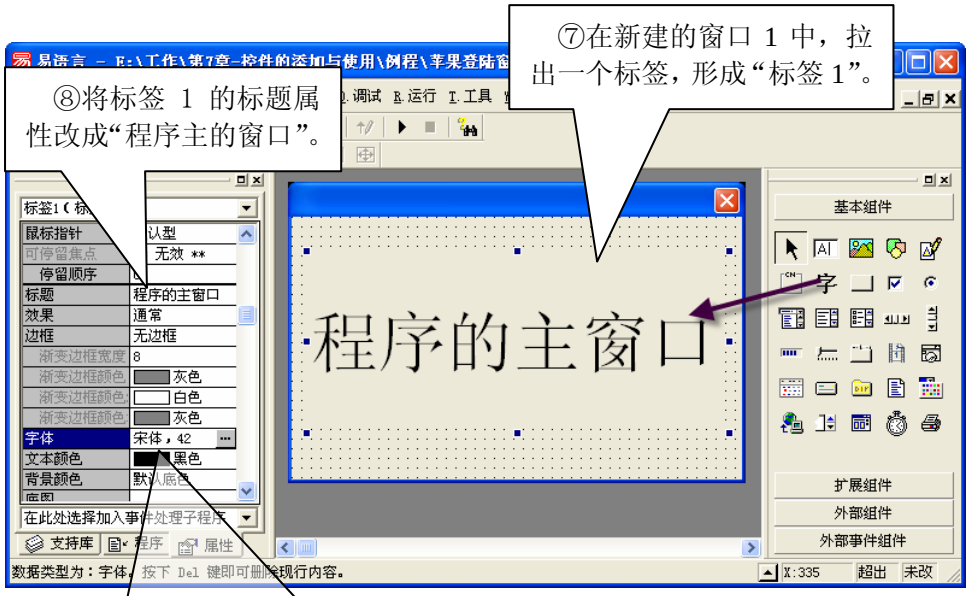
④将窗口的边框改回初始的普通固定边框。



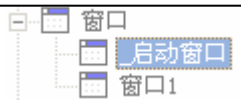
⑤将窗口的外形改成初始的矩形

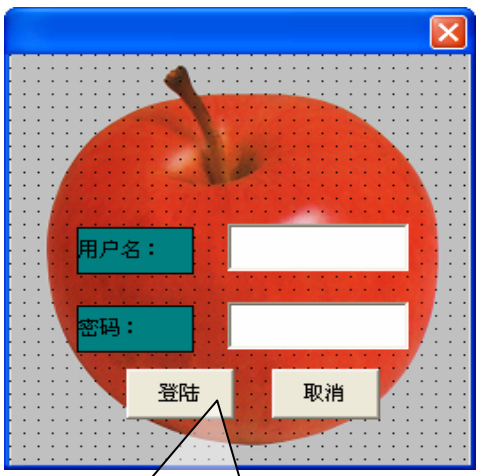


⑥在启动窗口上点击鼠标右键，在弹出菜单中选择“插入新窗口”，就会添加一个新的窗口。



双击启动窗口，回到启动窗口界面。





按钮被单击的事件是按钮最常使用的事件，运行程序后，只要按钮组件被单击就触发这个事件，并运行这个事件下的代码。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

双击按钮 1，就会产生“_按钮 1_被单击”的子程序。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

```
判断 (编辑框1.内容 = “123” 且
      编辑框2.内容 = “456”)
    _启动窗口.可视 = 假
    载入 (窗口1, , 真)
```

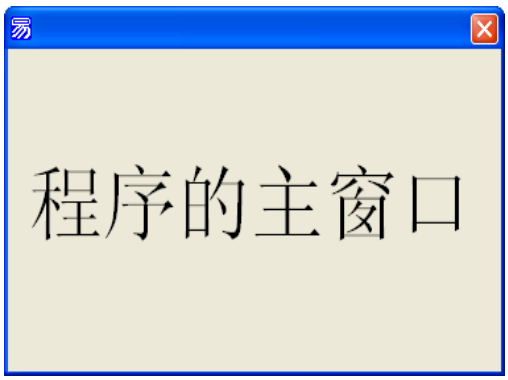
输入代码：

```
判断 (编辑框1.内容 = “123” 且 编辑框2.内容 = “456”)
```

判断编辑框 1 的内容是否等于“123”，并且，编辑框 2 的内容是否等于“456”，如果都满足条件，就运行判断中的代码。（即判断用户名和密码是否正确。）

当编辑框 1 内容等于“123”并且编辑框 2 的内容等于“456”的时候，就运行这 2 行代码，运行后，启动窗口的可视属性被设置成假，并载入窗口 1。

当组件的可视属性被设置成假以后，这个组件就处于隐藏状态 也就是看不到了。下面我们就按下 F5 键，来试运行上面的程序。



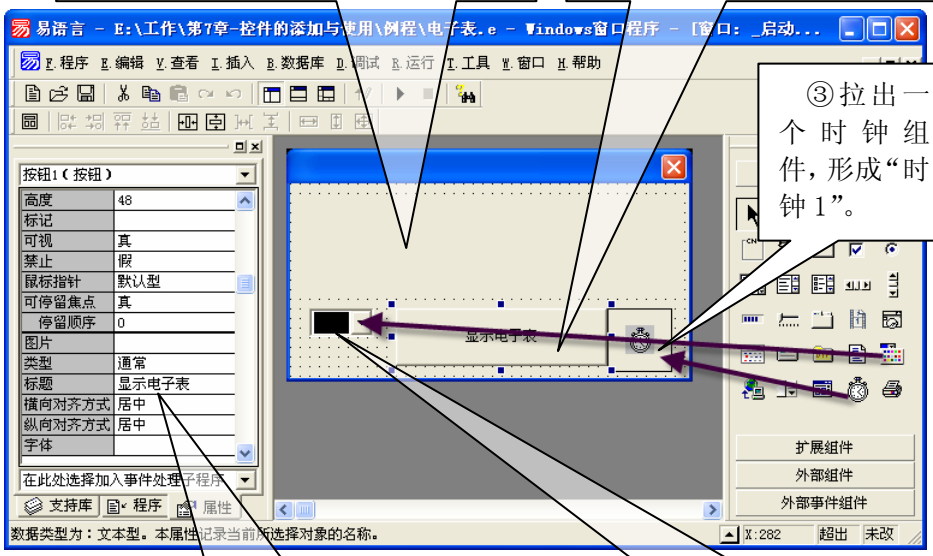
当输入正确的用户名和密码以后，点击“登陆”按钮，就会弹出窗口 1，而登陆窗口就消失不见了。

7.4 事件的应用—电子表

手表已经成为了生活中不可缺少的东西，而且现在有很多都是电子表，本章利用组件的各种事件，大家跟着图例自己动手做一个可以改变颜色的电子表。

①拉出一个标签组件，形成“标签1”，然后将它的标题属性清空。

②拉出一个按钮组件，形成“按钮 1”。



③拉出一个时钟组件，形成“时钟 1”。

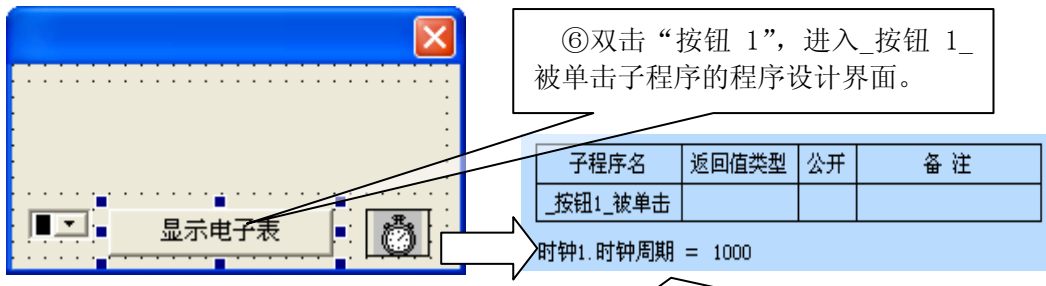
④将按钮 1 的标题改成“显示电子表”。

⑤拉出一个颜色选择器组件，形成“颜色选择器 1”。

时钟组件虽然过多的使用会占用很多系统资源，但它仍然是一个非常实用的组件，时钟组件是在规定的时钟周期里，不断的执行时钟组件独有的“周期事件”的子程序下的代码。所以，在使用时钟组件时，都要先规定时钟组件的时钟周期。例如：

```
时钟 1.时钟周期=1000
```

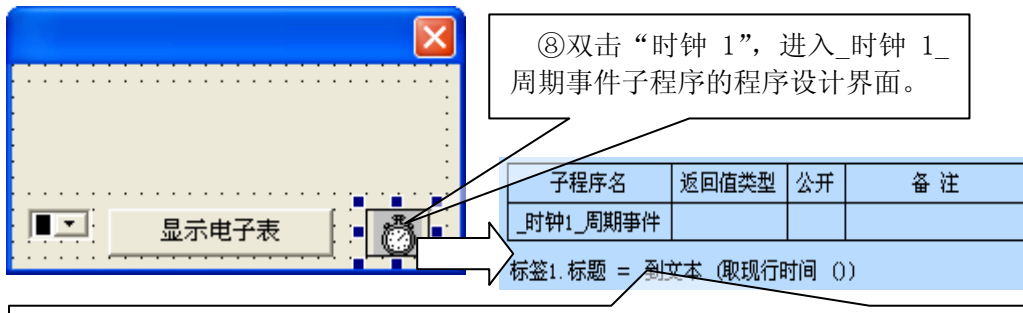
时钟周期规定 1000 就代表了每 1 秒钟执行一次周期事件子程序下的全部代码，依次类推，100 就代表了时钟周期 0.1 秒，10 就代表了时钟周期 0.01 秒。



⑦输入代码：

```
时钟1.时钟周期=1000
```

按下按钮后，就将时钟 1 的始终周期设置成 1000，表示时钟 1 已经开始运转了。



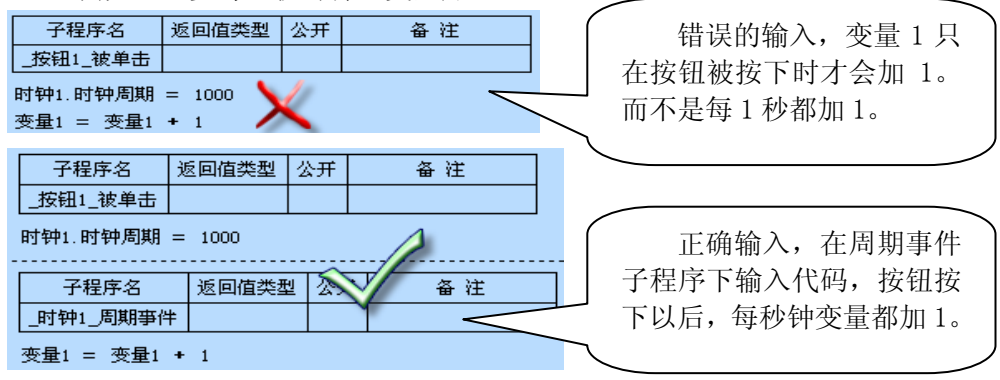
⑨输入代码：

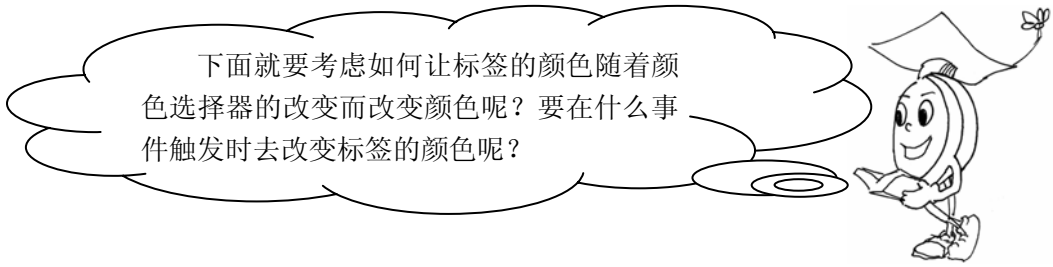
```
标签1.标题=到文本（取现行时间（））
```

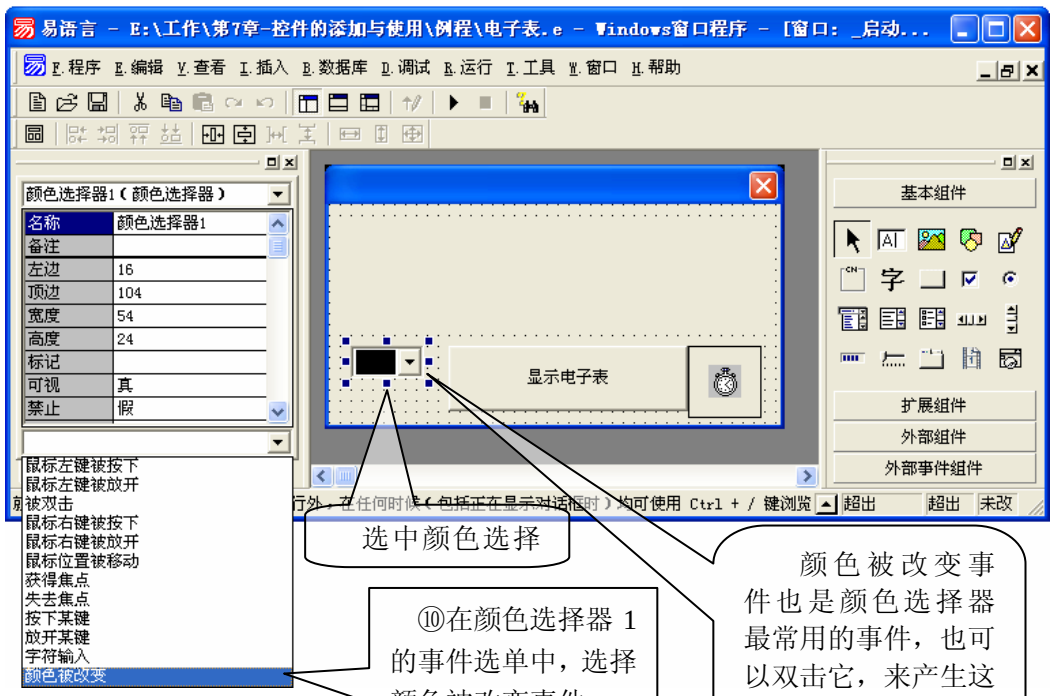
每个时钟周期都执行此行代码。即按下按钮 1 后就每 1 秒钟取一次现行时间。由于取现行时间命令返回的是日期时间型的数据，所以用到文本命令进行转换。

注意：想要在时钟的周期下执行代码，一定要在时钟组件的周期事件子程序下输入代码，不要在设置时钟组件的时钟周期时编写代码。

例如：想要每 1 秒钟都让变量加 1：







只要颜色选择器的颜色被改变，就执行这个子程序下的代码，即将标签的背景颜色改变。

子程序名	返回值类型	公开	备 注
_颜色选择器1_颜色被改变			

```
标签1.背景颜色 = 颜色选择器1.颜色
```

最后，在_颜色选择器 1_颜色被改变子程序下输入代码：

```
标签1.背景颜色 = 颜色选择器1.颜色
```

将标签的背景颜色改变成颜色选择器的颜色。

按下 F5 键运行一下制作好的电子表，并试着在颜色选择器中选择不同的颜色。

按下按钮，就显示出了当前的时间。

可以随意选择电子表颜色。

7.5 了解组件的方法

组件除了有属性、基本事件和自有事件外，有些组件还有自己的方法。组件方法的用法和命令的用法差不多，格式是：

组件名.方法名(参数)

组件的方法运行后就会对相应的组件产生不同的效果。

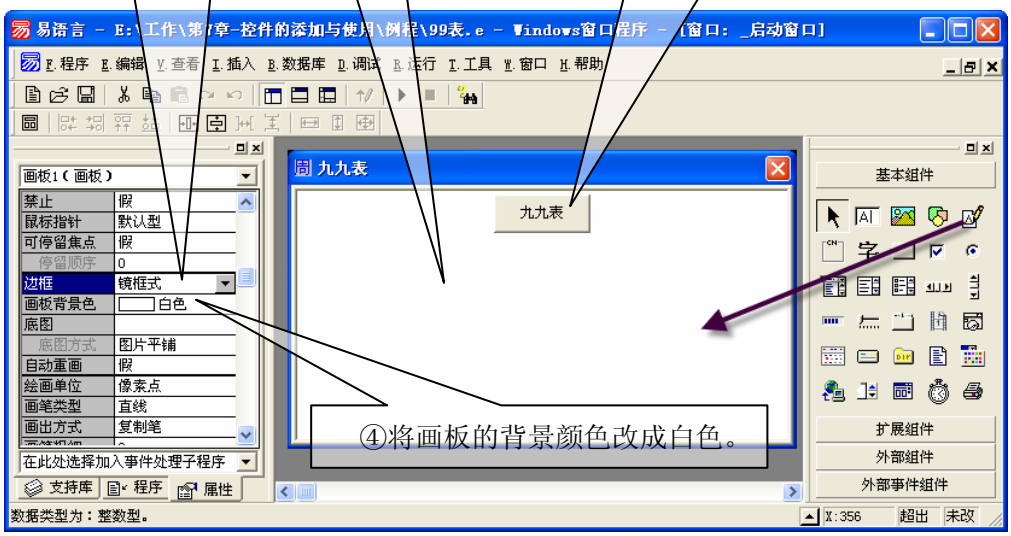
用画板组件就可以制作一个精美的小九九表。

下面先来设计一下小九九表的外观。

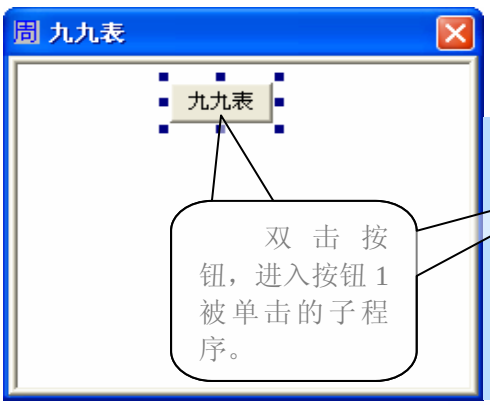
①拉出一个画板组件，形成“画板 1”。

②在画板上拉出一个按钮组件，形成按钮 1，并将按钮 1 的标题改成“九九表”。

③将画板的边框属性改成“镜框式”。



④将画板的背景颜色改成白色。



双击按钮，进入按钮 1 被单击的子程序。

⑤用[Ctrl+L]键，新建 3 个变量，并给变量起名和定义变量的类型。

子程序名	返回值类型	公开	备注
_按钮1_被单击			

变量名	类型	静态	数组	备注
乘号前	整数型			被乘数1－9
乘号后	整数型			乘数1－9
要显示的内容	文本型			

要想在画板上将九九表显示出来，首先考虑小九九表中有哪些数，他们有什么规律呢，只要把九九表中，每一个算式的乘数和被乘数依次取出，就可以得到要显示在画板上的内容，然后用画板组件的方法显示出来就可以了。

⑥在按钮 1 被单击的子程序下输入代码：

```
计次循环首 (9, 被乘数)
    变量循环首 (1, 被乘数, 1, 乘数)
        要显示的内容 = 到文本 (被乘数) + “×” + 到文本 (乘数) + “=” + 到文本 (被乘数×乘数 )
        画板1.定位写出 (乘数 × 40 - 30, 被乘数 × 20 - 10, 要显示的内容)
    变量循环尾 ()
计次循环尾 ()
```

变量名	类 型	静态	数组	备 注
被乘数	整数型			被乘数1－9
乘数	整数型			乘数1－9
要显示的内容	文本型			

```
计次循环首 (9, 被乘数)
    变量循环首 (1, 被乘数, 1, 乘数)
        要显示的内容 = 到文本 (被乘数) + “×” + 到文本 (乘数) + “=” + 到文本 (被乘数 × 乘数)
        画板1.定位写出 (乘数 × 40 - 30, 被乘数 × 20 - 10, 要显示的内容)
    变量循环尾 ()
计次循环尾 ()
```

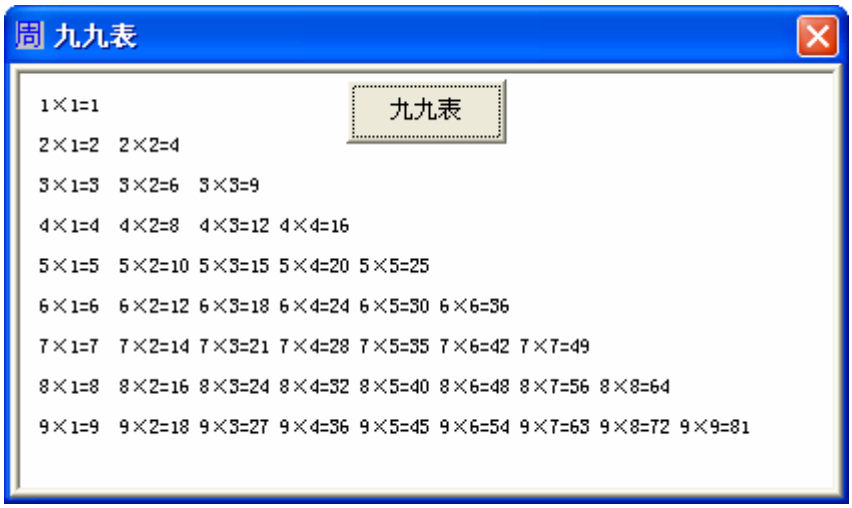
取出九九表中的的被乘数。

取出九九表中的乘数。

取出每个算式的文本。并保存在“要显示的内容”变量中。

用画板的定位写出的方法，将要“显示的内容”变量，显示在画板的指定位置上，定位写出方法的前两个参数规定了写出文本的坐标，是用一个公式表示的，只要了解定位写出方法的使用就可以了，公式仅做了解。

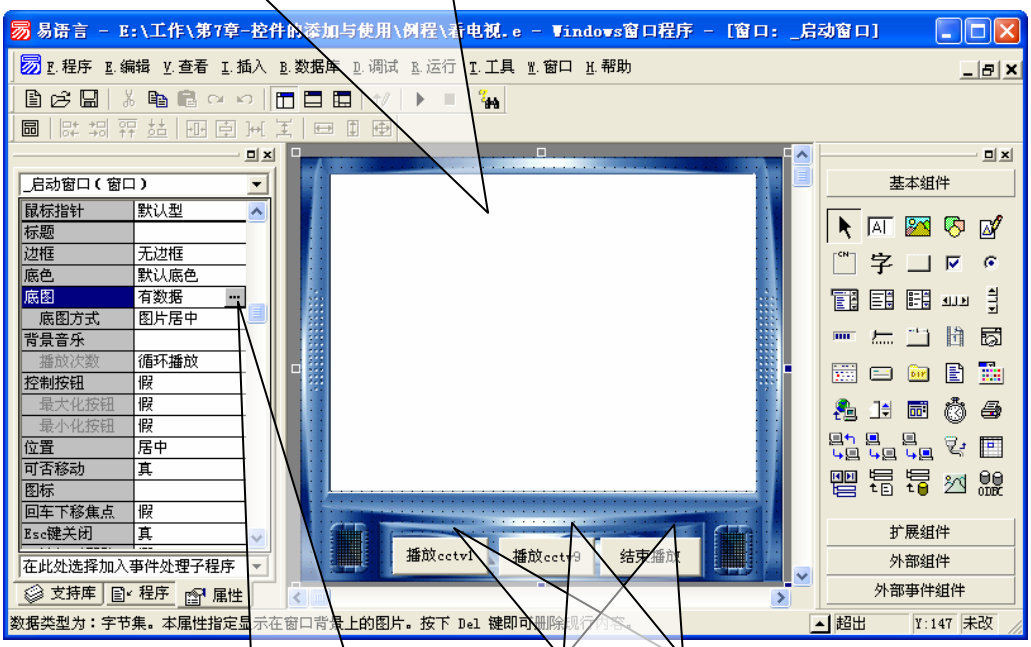
让我们按下 F5 键，运行一下九九表程序吧。按下“九九表”按钮，熟悉的九九表就显示出来了。是不是很神奇呢？



现在是网络时代了，看到众多的播放软件，都可以在网上直接观看电视节目，是不是也想自己动手做一个属于自己的网络电视呢。其实易语言可以用简单的代码，就能实现这个功能。

下面首先来设计网络电视的界面。

①创建一个新程序，并将启动窗口的边框属性设置成“无边框”。



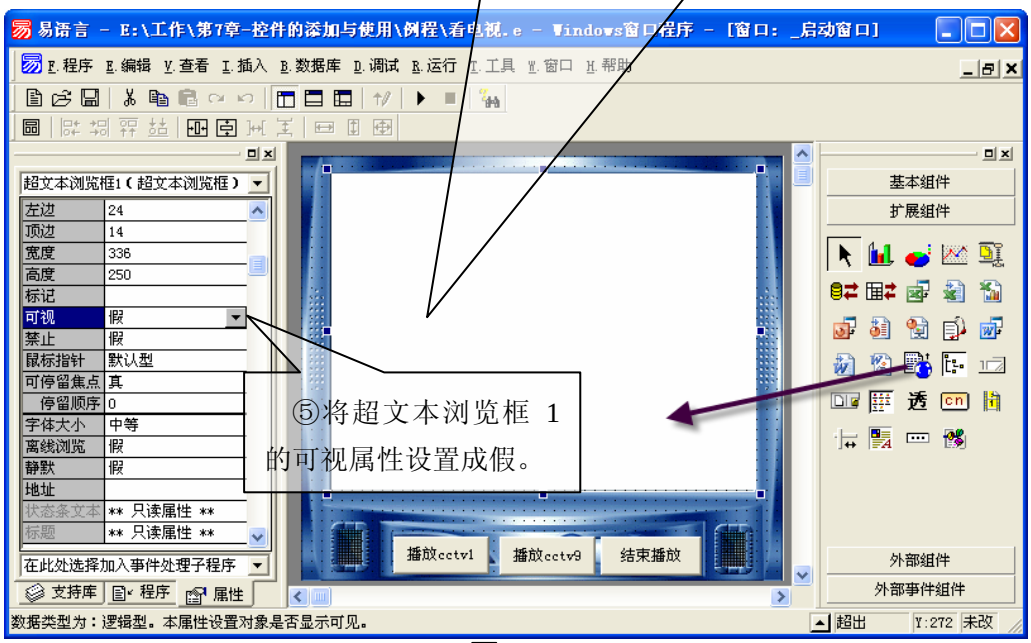
②点击底图属性上的小按钮。选择播放器的界面图片。

③拉出 3 个按钮组件，分别将标题属性改成“播放 cctv1”、“播放 cctv9”、“结束播放”。

选择一张图片，作为播放器的界面。

选好图片后，点击“打开”按钮。

④选择超文本浏览框组件，拉出一个和刚才窗体底图上黑色部分大小相同的超文本浏览框，形成了“超文本浏览框 1”。



双击“播放 cctv1”按钮，进入_按钮 1_被单击子程序下

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

超文本浏览框1.可视 = 真
超文本浏览框1.地址 = “http://www.cctv.com/prime/zxzb/cctv1.html”

按钮 1 被单击后就让超文本浏览框 1 的可视属性为真，即让超文本浏览框显示出来。

在“_按钮 1_被单击”子程序中输入以下程序代码：

```
超文本浏览框 1.可视 = 真
超文本浏览框 1.地址=
“http://www.cctv.com/prime/zxzb/cctv1.html”
```

这个超文本浏览框浏览网址：http://www.cctv.com/prime/zxzb/cctv1.html
该网址是中央电台在线直播的网址。大家可以直接在 IE 浏览器中输入这个网址。

然后按照编写“播放 cctv1”按钮的方法，回到启动窗口，双击“播放 cctv9”按钮，和“结束播放按钮”，然后在相应的子程序下输入代码。

在双击“播放 cctv9”按钮后产生的子程序下输入代码：

```
超文本浏览框1.可视 = 真
超文本浏览框1.地址 =
“http://www.cctv.com/prime/zxzb/cctv9.html”
```

这个子程序下输入的代码和_按钮 1_被单击子程序下输入的代码是相同的，只是将网址改成了中央 9 台的直播网址。

子程序名	返回值类型	公开	备 注
_按钮2_被单击			

超文本浏览框1.可视 = 真

超文本浏览框1.地址 = “http://www.cctv.com/prime/zxzb/cctv9.html”

子程序名	返回值类型	公开	备 注
_按钮3_被单击			

结束 ()

在双击“结束播放按钮”后产生的子程序下输入代码：

```
结束 ()
```

用来关闭播放器

一个精美的 “网络电视播放器”就制作完成了，可以在线收看中央 1 台和中央 9 台的节目了，大家也可以收集更多的网址，让这个网络电视机收到更多的台，但要提醒大家的是，这个播放器是要在安装了“Windows Media Player9”以后才可以正常使用的，快去：

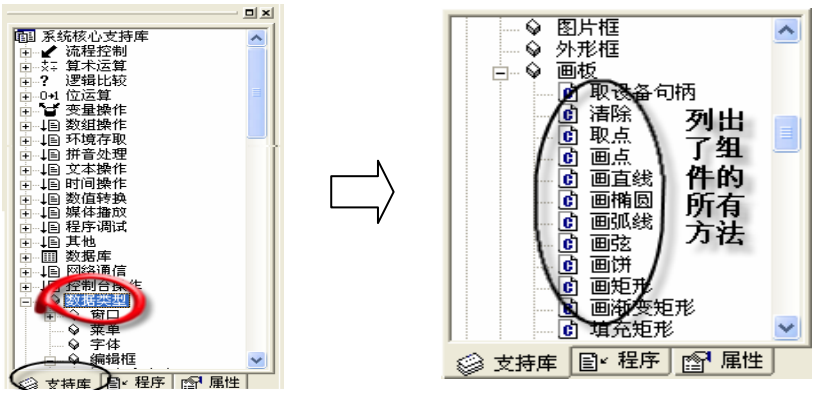
http://www.microsoft.com/windows/windowsmedia/download/下载吧。

然后按下 F5 键试运行刚才编写的网络电视。

用简单的几行代码就可以实现播放网络电视的功能，是不是很有意思呢？

7.7 课后练习

（1）对每个组件是属性，事件和方法做详细了解，观察哪些属性是某组件的子有属性，哪些事件是某组件的子有事件，组件的方法都有哪些：



（2）不用任何代码，制作一个万年历，想一想，用什么组件就可以实现，效果如下：

（3）动手做一个可以旋转的字。提示：

1 使用时钟组件

2 使用字体属性中的字体角度属性，表示为：组件名.字体.字体角度=？？

效果如下：

第 8 章 易语言的子程序

本章主要介绍“易语言”子程序的调用方法、子程序参数的使用方法以及参数属性的相关使用方法。

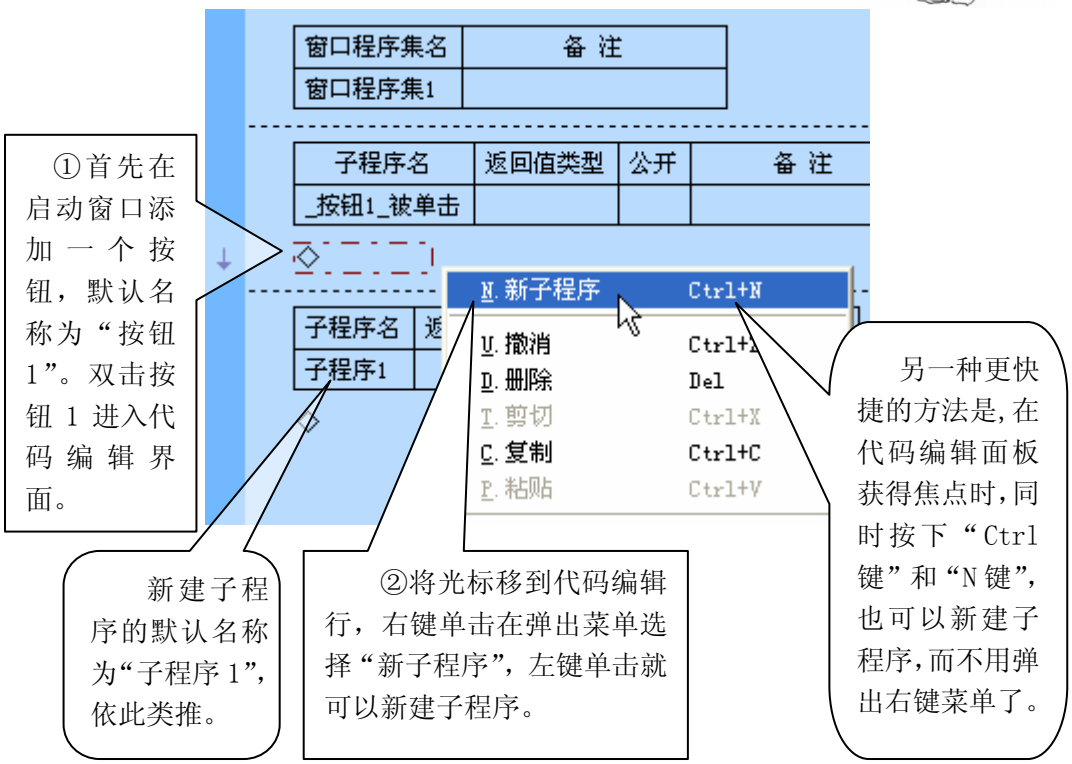
本章学习内容：

将程序分割成较小的逻辑单元就可以简化程序设计任务，这些部件被称为子程序。 子程序可用于压缩重复任务或共享任务，例如，压缩频繁的计算处理等等。

- 子程序可使程序划分成离散的逻辑单元，每个单元都比无子程序的整个程序容易调试及理解；
- 一个应用程序中的子程序，往往不必修改或只需稍作改动，便可以成为另一个程序的子程序。

8.1 新建子程序

下面我们就开始新建第一个自定义子程序。方法十分简单，一步一步跟我来吧！

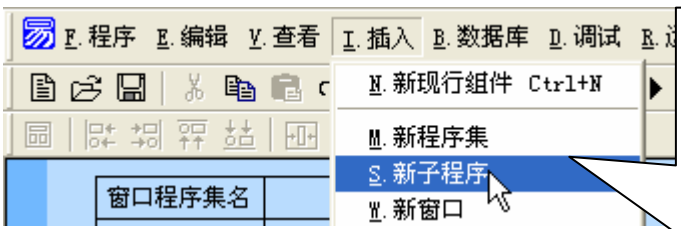


①首先在启动窗口添加一个按钮，默认名称为“按钮1”。双击按钮1进入代码编辑界面。

新建子程序的默认名称为“子程序1”，依此类推。

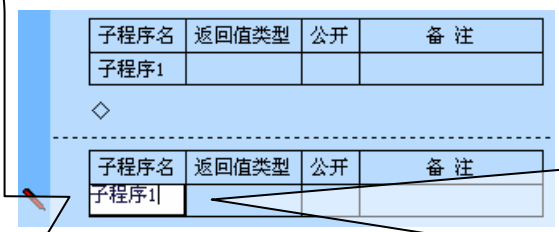
②将光标移到代码编辑行，右键单击在弹出菜单选择“新子程序”，左键单击就可以新建子程序。

另一种更快捷的方法是，在代码编辑面板获得焦点时，同时按下“Ctrl键”和“N键”，也可以新建子程序，而不用弹出右键菜单了。



第三种方法，到主菜单上。选择“插入”，在弹出菜单中选择“插入”，在弹出菜单中选择“新子程序”也可新建子程序。

注意：在同一个程序集中，子程序名不能重复。特别在修改的时候，“易语言”会提示并修改已经被使用过的名称。



“子程序1”已经存在，当另一个子程序被修改成“子程序1”时，“易语言”弹出信息框提示。

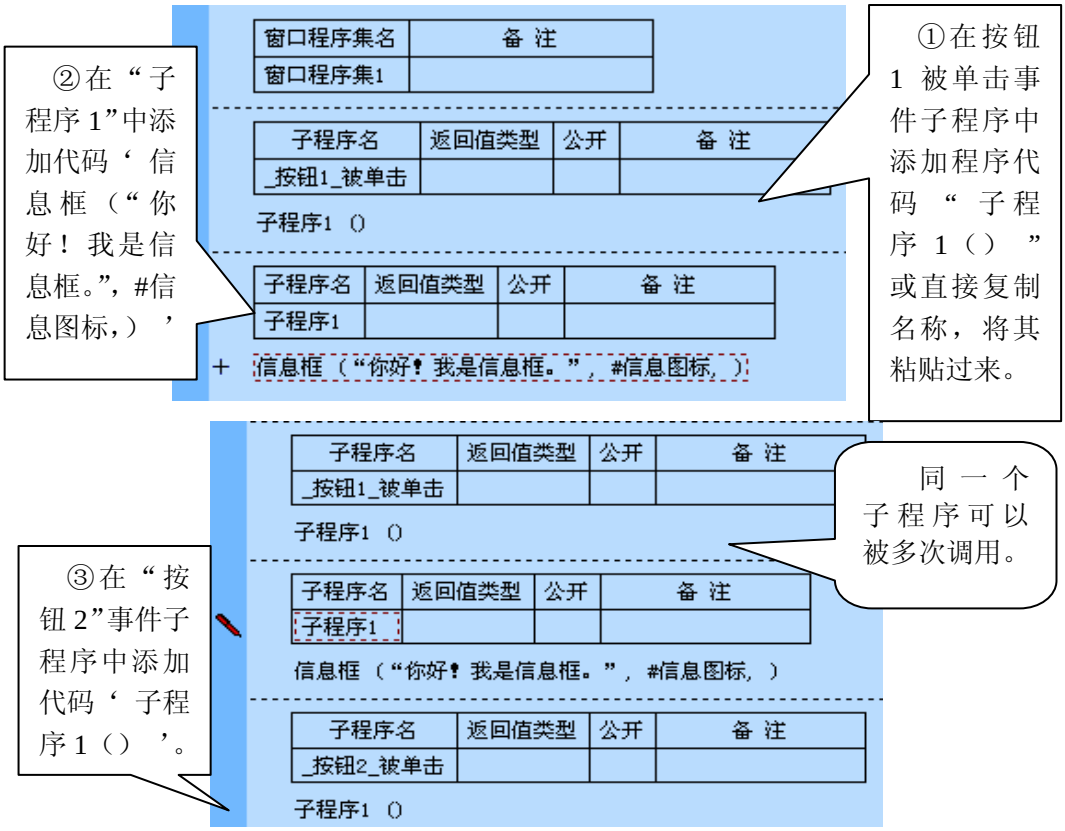
被“易语言”修改的子程序名。

新建的子程序没有参数和变量，这需要操作者自行添加和自定义参数的数量以及每个参数的类型、可空、参考、数组属性。

特别应该注意的是，每个新建的子程序都不会被“易语言”或其它子程序自动调用，下面将讲解如何调用子程序。

8.2 调用子程序

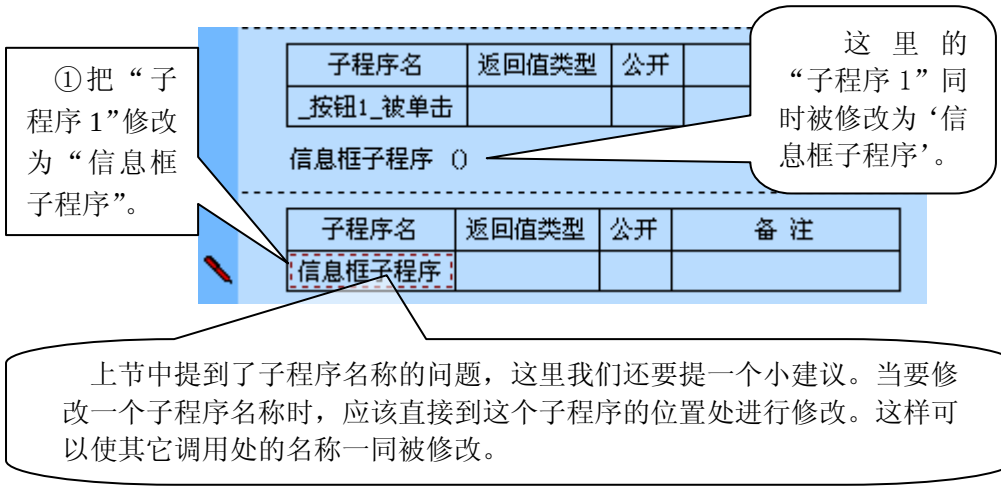
前一节中讲了如何建立子程序，下面演示如何调用一个简单的子程序。



④按“F5键”运行程序，分别单击“按钮1”和“按钮2”，都能弹出“子程序1”里的信息框。

子程序可以在其它事件子程序和新建子程序中多次被调用，也可以被其它程序集的事件子程序和新建子程序调用，就像单击“按钮1”，“_按钮1_被单击”被调用。要注意的是，子程序只能在本程序中使用。

每次调用子程序时，子程序中的所有语句都将被从第一条开始顺序执行，当执行到子程序尾部或者遇到“返回”命令时即返回到调用此子程序语句的下一条语句处。



子程序必须由其它事件子程序调用。子程序的调用方法与命令的调用方法完全一致。所谓事件子程序，就是“易语言”组件自带的触发事件。如：按钮被按下。

8.3 与事件子程序的区别

虽然新建子程序和事件子程序的调用方法都一样，但它们之间也有一些不同之处。

这是按钮 1 被单击子程序的原型，没有参数。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

①光标选中子程序名称，按“Enter 键”生成一个参数。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			
参数名	类 型	参考	可空 数组
参数1	整数型		

②添加参数名称并设置类型。最后按“F5 键”运行程序，单击“按钮 1”，信息框不见了。

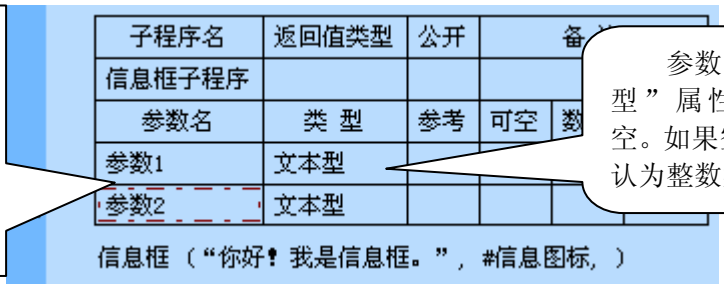
信息框看不到的原因是“_按钮 1_被单击”子程序参数结构被改变，“易语言”就会把它当作新建子程序，和“按钮 1”没有任何关系了，而新建子程序可以任意的添加参数。

8.4 子程序参数

参数和变量的使用方法一样，可以使用赋值语句在参数内临时存 储数据。参数有“**名字**”（用来引用参数所包含内容的词）、“**数据类型**”（确定参数能够存储数据的种类）、“可空”和“参考”。参数只可以在被定义的子程序中使用，相当于局部变量。

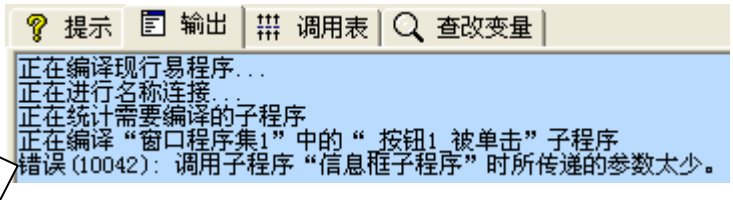
①把 8.3 中“_ 按钮 1_被单击”事件子程序的“参数 1”删除。

②在“信息框子程序”添加两个文本型参数“参数 1”和“参数 2”。

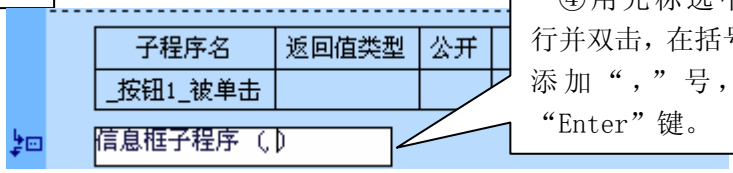


参数的“类型”属性可为空。如果空，默认为整数型。

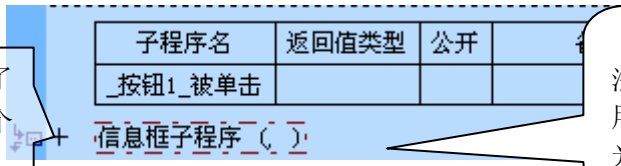
③按“F5”键，程序无法运行，在输出面板中可以看到次行输出文本。这就说明我们还要在调用处提供具体的参数。



④用光标选中此行并双击，在括号内添加“，”号，按“Enter”键。



⑤看到了吗？多了个“＋”号，我们可以将它展开。



展开的方法：直接双击或用键盘上的右光标键。

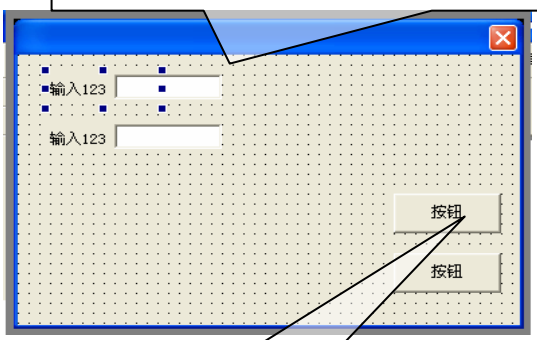
它们是互相对应的，在上面添加文本型数据“ 易语言”，当程序运行时，下面“参数 1”保存的数据就是“易语言”。

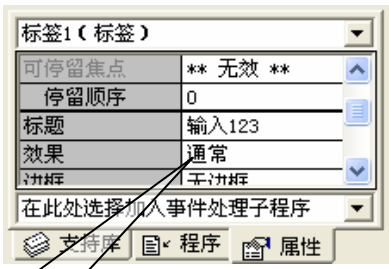
子程序名	返回值类型	公开	备 注
_按钮1_被单击			

信息框子程序 ( )
※参数1：
※参数2：

子程序名	返回值类型	公开	备 注		
信息框子程序					
参数名	类 型	参考	可空	数组	备 注
参数1	文本型				
参数2	文本型				

①接下来我们在程序中为子程序提供具体的参数数据。用“Ctrl+Tab”键切换到界面设置区，在窗体上添加两个标签和两个编辑框，并将编辑框的标题设置为“输入 123”，提示用户。





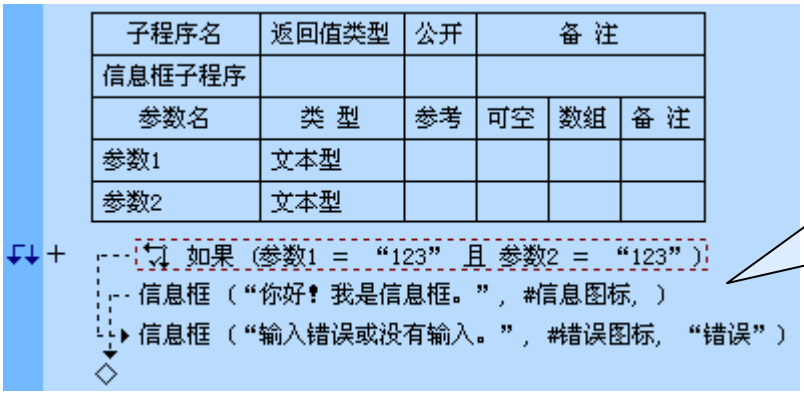
这是前面提到的“按钮 1”，下面的是“按钮 2”。

标签的标题在此处设置。

②切换到代码编辑面板，在子程序调用处，分别添加数据(编辑框提供的文本型数据)。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

信息框子程序 (编辑框1.内容, 编辑框2.内容)
※参数1： 编辑框1.内容
编辑框2.内容



③ 修改“信息框子程序”代码。

④按“F5”运行程序，在“编辑框 1”和“编辑框 2”中输入“123”，单击“按钮 1”，第一个信息框被弹出。你也可以输入其它的数据，看一看效果。

子程序如需要接收参数数据，必须先在子程序定义表中参数表部分定义与欲接收数据数目相同的参数。调用子程序时所传递过来的数据将被顺序地填入对应的参数中。如果所传递过来的数据与对应位置处的参数数据类型不一致，在可以互相转换时，系统将自动进行转换，否则会产生运行时错误。

1、子程序可以接收参数，所定义的各参数的数据类型及参数数目决定了该子程序所能够接收的参数数据的类型和数目，具有参数的子程序在被调用时必须提供与参数数目相同的数据。如上面的第 2 步在调用子程序的同时就把**编辑框 1. 内容**和**编辑框 2. 内容**作为参数数据对应地传递到了**编辑框子程序**内的**参数 1**、**参数 2** 参数内；

2、参数仅能在子程序内部使用，使用方法等同于变量。

8.5 参数的可空属性

参数可空，从表面上就能理解它的意思。就是在调用子程序时，可以不提供在子程序中被定义成可空的参数任何数据。看看下面。

子程序名	返回值类型	公开	备 注		
信息框子程序					
参数名	类 型	参考	可空	数组	备 注
参数1	文本型				
参数2	文本型		✔		

```
如果 (参数1 = “123” 且 参数2 = “123”)
    信息框 (“你好！我是信息框。”, #信息图标, )
    信息框 (“输入错误或没有输入。”, #错误图标, “错误”)
```

①将“参数2”可空属性选中。方法是，选中单击或选中按“空格”键。

②把调用处的“参数 2”删除。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

```
信息框子程序 (编辑框1.内容, )
    ※参数1： 编辑框1.内容
```

如果本属性为真，那么在调用本子程序时，调用方可以不为此位置处的参数传递数据。主要用作支持具有默认值的参数，也可以在为子程序添加了新参数后又不想去更改以前调用此子程序的语句时使用。

8.6 参数的参考属性

打个比方，你家的门锁有两把钥匙，分别被两个人拿着。不管是哪一个人都可以打开房门，改变家里家具的摆放位置。参考就是在调用一个子程序之前设置一个提供参数数据的变量的值，在子程序中相对应的参数值被改变，调用后这个变量值同时被改变。

①在按钮 1 被单击事件子程序中添加两个局部变量。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
变量1	整数型			
变量2	整数型			

```
变量1 = 到数值 (编辑框1.内容)
变量2 = 到数值 (编辑框2.内容)
信息框子程序 (变量1, 变量2)
信息框 ("参考的使用方法。" + #换行符 + "变量2==" + 到文本 (变量2), 0, )
```

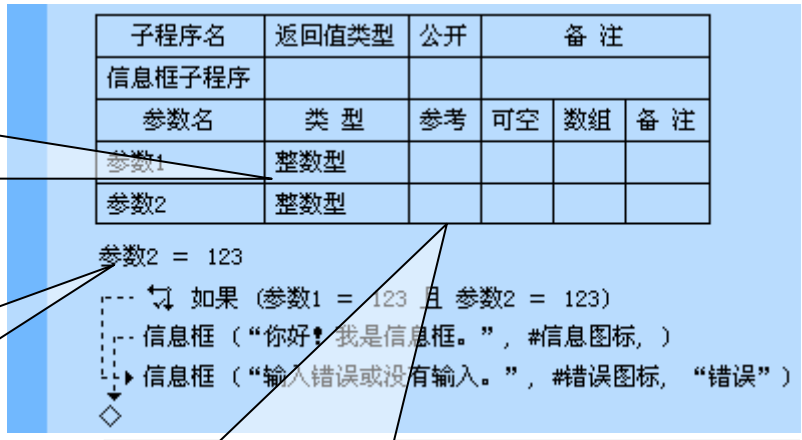
②把编辑框提供的文本型数据转换到整数型。

注意：

由于文本型数据默认被参考，看不到参考的效果，所以把文本型数据转换到整数型。

③添加信息框，显示调用子程序后，“变量 2”的值。

④将“信息框子程序”两个参数的类型改为整数型。



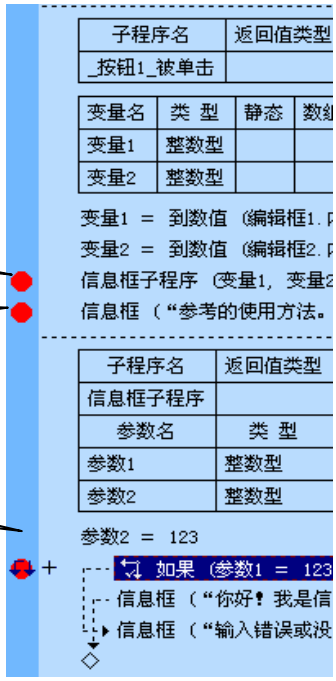
⑤添加代码，修改“参数 2”的值。

⑥设置“参数 2”的参考属性。像可空属性设置的方法一样，在这里设置参考。

注意：

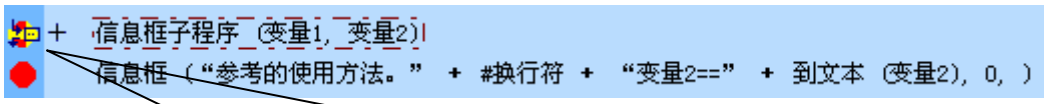
为了更好的了解程序的运行过程，可以通过在关键代码前设置断点的方法知道代码行被执行的情况。

⑦在代码前设置断点。

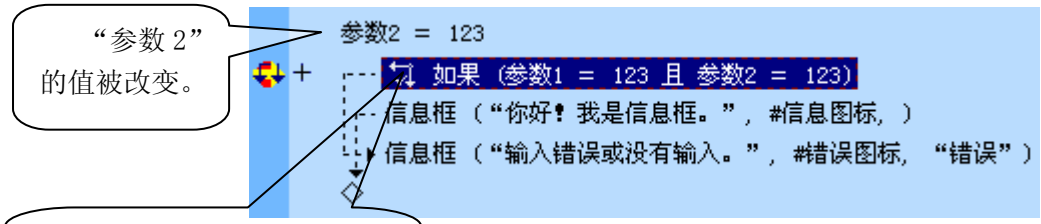


设置断点的方法：用光标选中将被设置断点的代码行，按“F9”键。

⑧按“F5”运行程序，在启动窗口的编辑框中输入“123”，单击“按钮 1”。

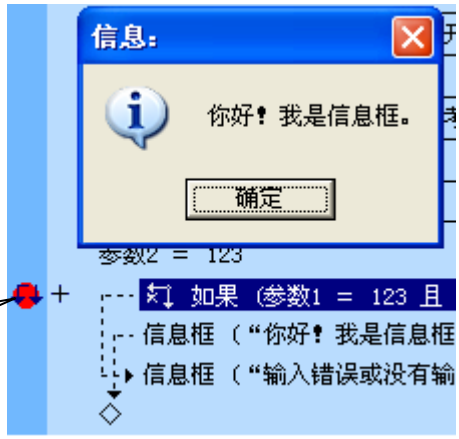


窗口消失，程序停止，一个断点变黄。这就说明在按钮 1 被单击的事件子程序中，第一个断点处的“信息框子程序”被执行。



“参数 2”的值被改变。

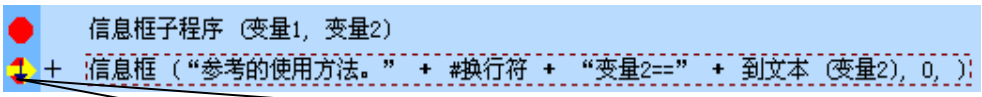
“参数 1”的值是“变量 1”的值。两个条件都成立。



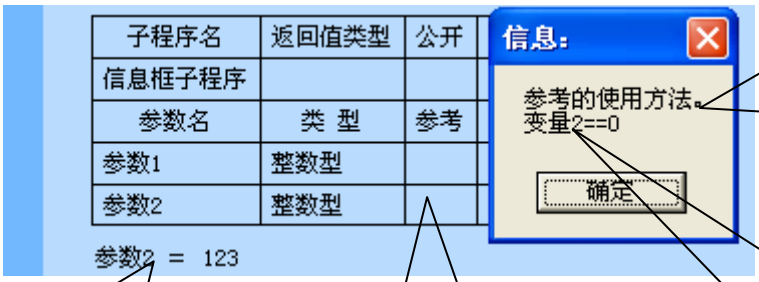
再按“F5”键，“信息框子程序”里的代码被执行。说明子程序已经被调用。

再按“F5”键，断点变为红色，弹出条件成立的信息框。

⑨单击“确定”按钮。



程序又跳到了“信息框子程序”被调用处的下一行代码处，说明“信息框子程序”执行完毕。



再按“F5”键，弹出按钮 1 被单击事件子程序中的信息框。

即使“参数 2”的数据被改变，“变量 2”仍保存原值。

“参数 2”没有设置参考。

由于运行开始时“编辑框 1”内的数据为空文本，转换到整数后为“0”。

⑩ 单击“确定”按钮，代码运行结束。关闭程序，设置“参数 2”的参考，按“F5”键运行。最后信息框显示“变量 2”的值和“参数 2”的值相同。

设置系统为当前子程序参数传递数据时是否为传递指向数据的指针。如果所传递过来的参数数据为**数组、用户定义数据类型、库定义数据类型、字节集型、文本型数据**，则无论此属性是否为真，都将传递指针。如果所传递过来数据的类型与相应位置处参数的数据类型不一致但可以相互转换，譬如将“整数型”数据传递到“小数型”的参数中，则在数据被实际传递前，系统将首先自动将“整数型”数据转换为“小数型” 数据，然后再进行传递。因此在这种情况下，即使本属性为真，系统也无法传递指向原参数数据的指针，只能传递数据本身。如果系统将数据指针成功地传递过来，那么在子程序中对此参数的内容的更改将会相应地反映到调用子程序时所提供的参数数据上。

8.7 子程序的返回值

子程序可以返回数据，但必须首先定义返回数据的类型，并且在程序中使用“返回”命令进行返回。

①将“信息框子程序”的返回值类型设置为整数型。

子程序名	返回值类型	备 注			
信息框子程序	整数型				
参数名	类 型	参考	可空	数组	备 注
参数1	整数型				
参数2	整数型				

②删除下面两行代码。

```
参数 2 = 123

信息框 (“参考的使用方法。” + #换行符 + “变量 2==” + 到文本 (变量 2), 0,)
```

变量名	类 型	静态	数组	备 注
变量1	整数型			
变量2	整数型			
返回值	整数型			

③在按钮 1 被单击事件子程序中添加名称为“返回值”的整数型变量。

④“返回值”变量保存“信息框子程序”返回的整数型数值。

```
变量1 = 到数值 (编辑框1.内容)
变量2 = 到数值 (编辑框2.内容)
返回值 = 信息框子程序 (变量1, 变量2)
```

⑤在“信息框子程序”的每个分支处添加返回命令。

```
如果 (参数1 = 123 且 参数2 = 123)
    返回 (1)
    返回 (0)
```

⑥将“信息框子程序”里的两个信息框复制到按钮 1 被单击子程序的判断语句中。

⑦最后按“F5”运行。

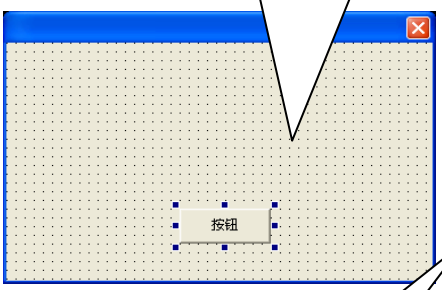
```
判断 (返回值 = 1)
    信息框 (“你好！我是信息框。”, #信息图标, )
    信息框 (“输入错误或没有输入。”, #错误图标, “错误”)
```

每次调用子程序时，子程序中的所有语句都将被从第一条开始顺序执行，当执行到子程序尾部或者遇到“返回”命令时即返回到调用此子程序语句的下一条语句处。

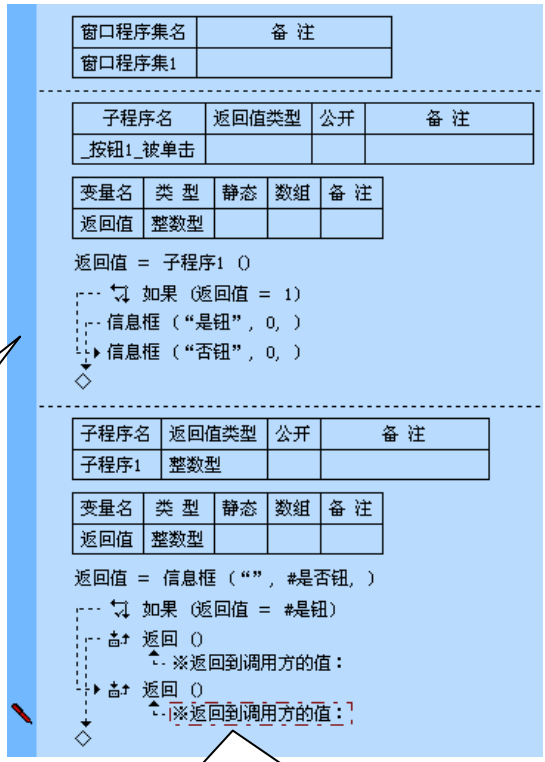
8.8 课后练习

（1）练习子程序的建立与调用。

①新建“Windows 窗口程序”添加一个按钮。

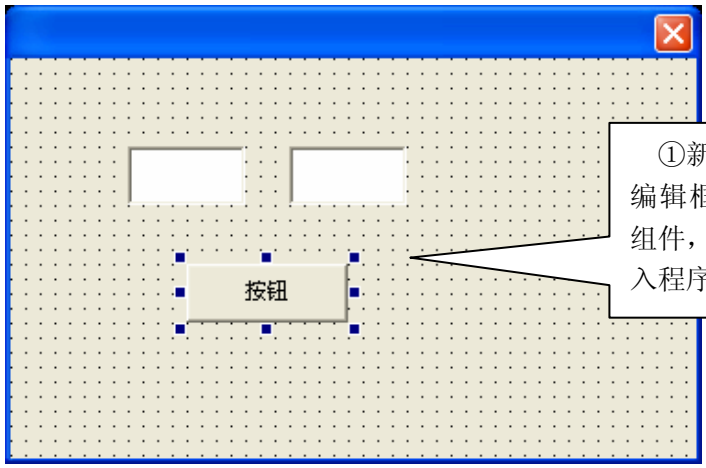


②添加如图上的代码。完成试运行一下。



这两个地方添加什么值，才可以使程序运行，并且能够正确显示。

（2）练习编写用子程序的方式表示两数相加，返回和的例程。



①新建易程序，放两个编辑框组件与一个按钮组件，双击按钮组件，进入程序代码界面。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

信息框 (和是几 (到数值 (编辑框1.内容), 到数值 (编辑框2.内容)), 0,)

子程序名	返回值类型	公开	备 注		
和是几	整数型				
参数名	类 型	参考	可空	数组	备 注
数1	整数型				
数2	整数型				

返回 (数1 + 数2)

②新建一个子程序，名称为：“和是几”。加入两个数。参数类型为整数型。返回值类型为整数型。

③在“_按钮 1.被单击”子程序下输入以下程序代码：

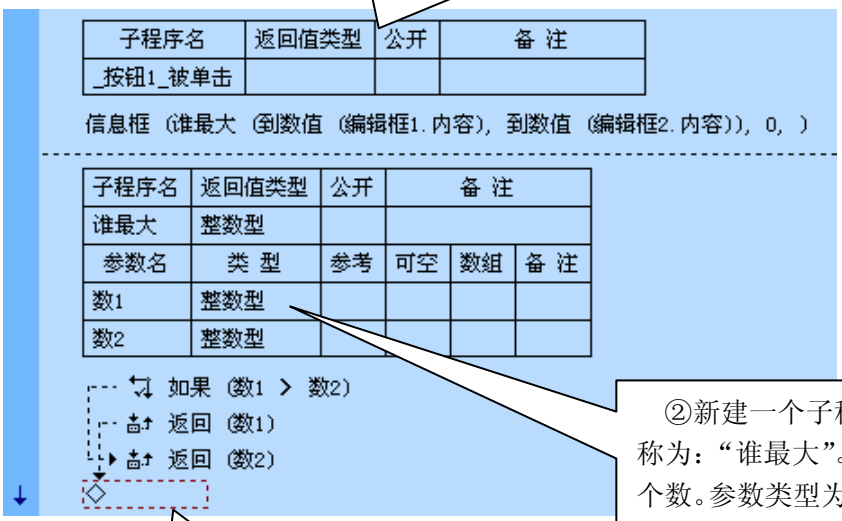
```
信息框 (和是几 (到数值 (编辑框 1.内容), 到数值 (编辑框 2.内容)), 0, )
```

在“和是几”子程序下输入以下程序代码：

```
返回 ( 数 1 + 数 2 )
```

（2）用子程序的方式表示判断两个数谁大，并返回最大数。

①开始也同上一题一样，也是新建易程序，放两个编辑框组件与一个按钮组件，双击按钮组件，进入程序代码界面。



②新建一个子程序，名称为：“谁最大”。加入两个数。参数类型为整数型。返回值类型为整数型。

在“_按钮 1.被单击”子程序下输入以下程序代码：

```
信息框 (和是几 (到数值 (编辑框 1.内容), 到数值 (编辑框 2.内容)), 0, )
```

在“和是几”子程序下输入以下程序代码：

```
如果 (数 1 > 数 2)
    返回 (数 1)
否则
```

第 9 章　易语言的易模块

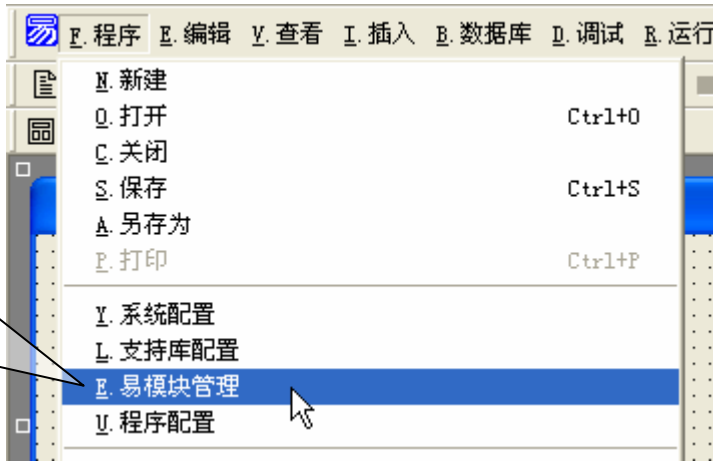
本章主要介绍“易模块”的安装、使用方法以及新建、保存的方法。同时介绍一个非常简单的“易模块”编写过程，以帮助用户了解和学习。

本章学习内容：

9.1 “易模块”的安装

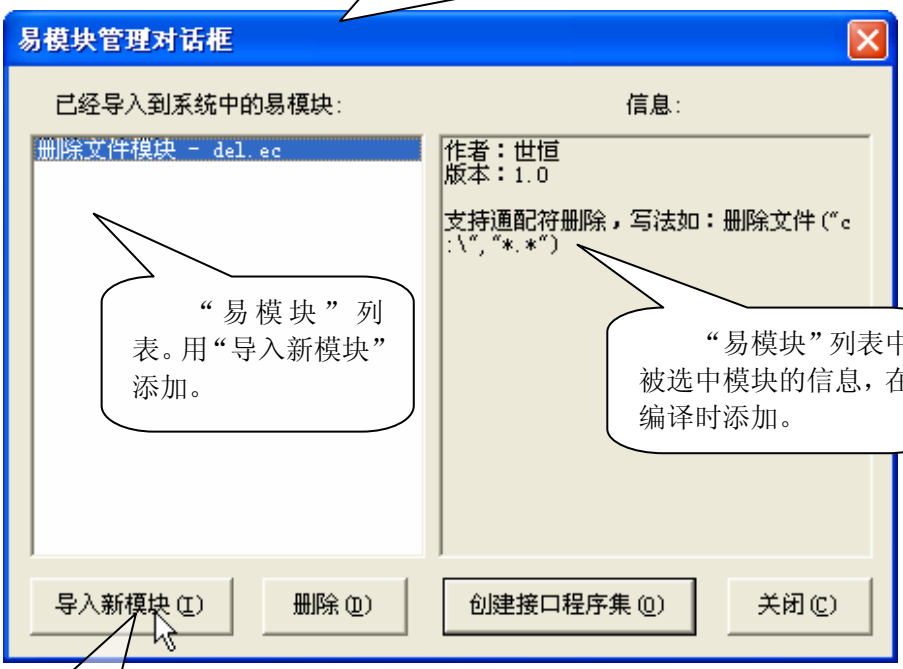
大家先学习别人写好的易模块，再自己动手制作一个易模块吧。

易模块文件的后缀是“.ec”。为了能够了解“易模块”的使用方法，首先要正确安装。



①单击“易语言”系统菜单上的“程序”→“易模块管理”。

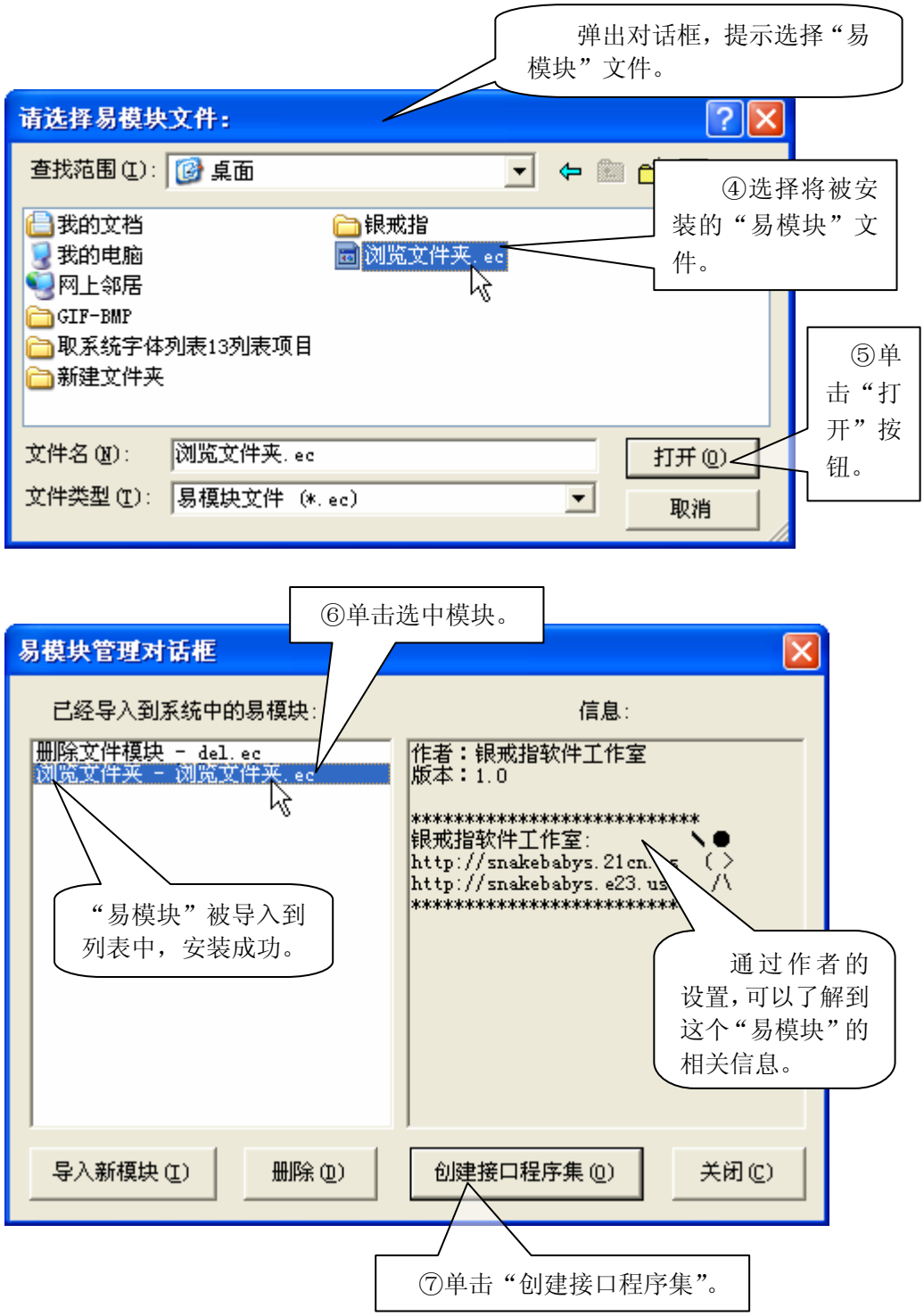
弹出易模块管理对话框。用于管理易模块，如导入、删除、创建接口程序集。

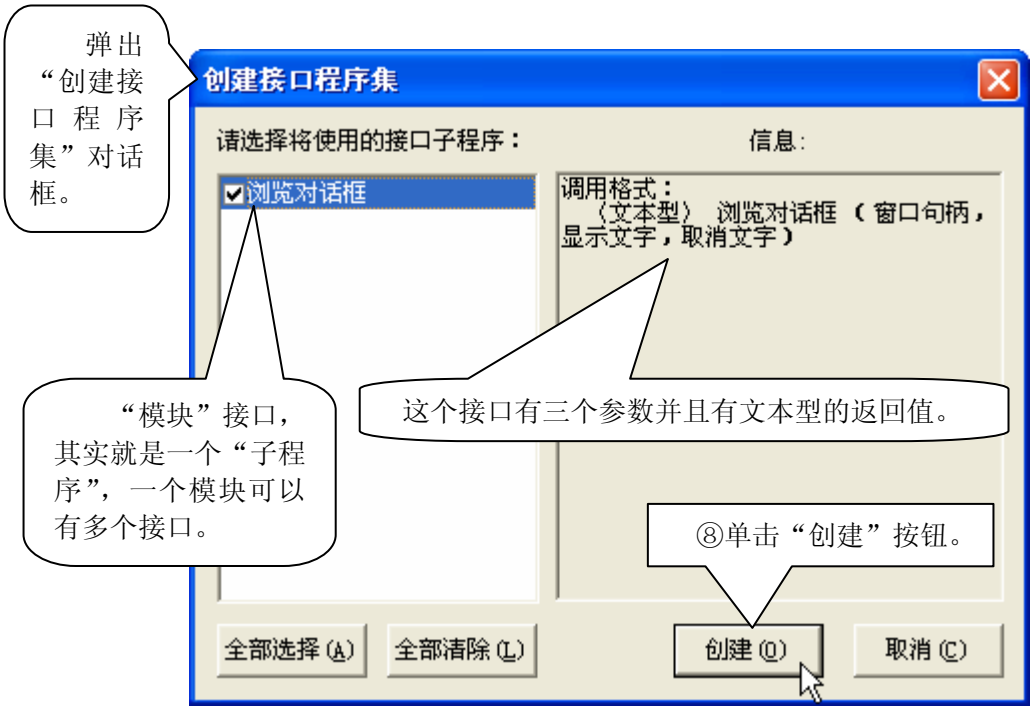


“易模块”列表。用“导入新模块”添加。

“易模块”列表中被选中模块的信息，在编译时添加。

②单击“导入易模块”按钮。


弹出对话框，提示选择“易模块”文件。
请选择易模块文件：
查找范围(I): 桌面
我的文档
我的电脑
网上邻居
GIF-BMP
取系统字体列表13列表项目
新建文件夹
银戒指
浏览文件夹.ec
④选择将被安装的“易模块”文件。
文件名(N): 浏览文件夹.ec
文件类型(T): 易模块文件 (*.ec)
打开(O)
取消
⑤单击“打开”按钮。
⑥单击选中模块。
易模块管理对话框
已经导入到系统中的易模块:
信息:
删除文件模块 - del.ec
浏览文件夹 - 浏览文件夹.ec
作者：银戒指软件工作室
版本：1.0
****
银戒指软件工作室:
http://snakebabys.21cn.
http://snakebabys.e23.us
“易模块”被导入到列表中，安装成功。
通过作者的设置，可以了解到这个“易模块”的相关信息。
导入新模块(I)
删除(D)
创建接口程序集(O)
关闭(C)
⑦单击“创建接口程序集”。



模块的程序集名称，也是模块的文件名称。前面被自动加上“_模块_”标记，提示为模块。

在代码编辑面板生成接口程序集。

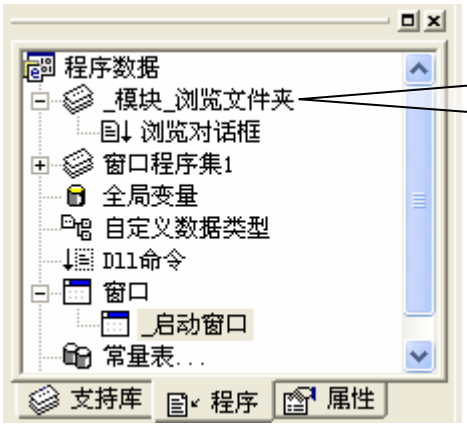
程序集名	备 注
_模块_浏览文件夹	** 不要更改此处 浏览文件夹.ec

子程序名	返回值类型	公开	备 注		
浏览对话框	文本型				
参数名	类 型	参考	可空	数组	备 注
窗口句柄	整数型	✔			所在窗口句柄，注意不是控件句柄
显示文字	文本型	✔			
取消文字	文本型	✔			放弃选择时的文字

※备注：** 本子程序功能由系统自动转交对应模块实现，可以删除但不能修改。

三个参数

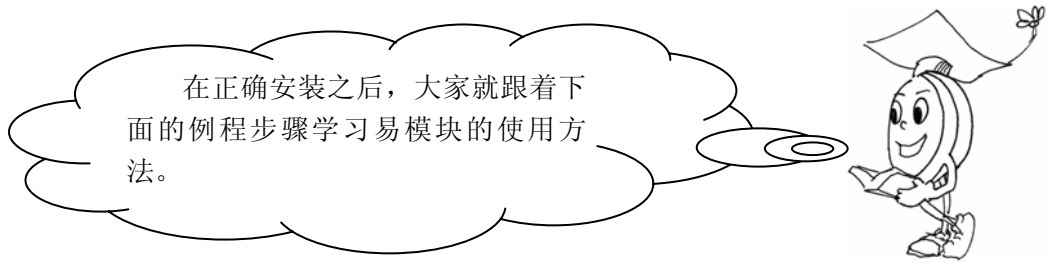
文本型的返回值。

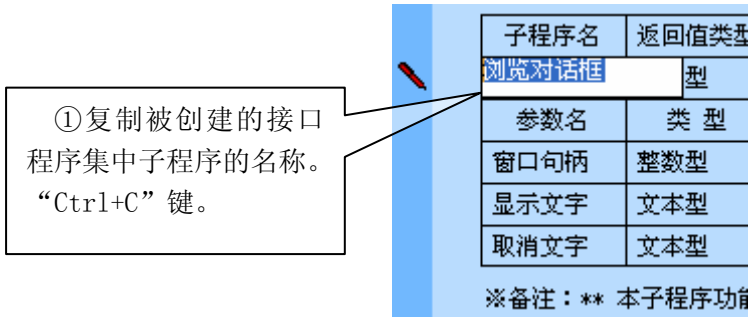


注意：

用“易语言 3.5”以前版本开发的个别“易模块”需要通过原作者的修改，才能在“易语言 3.6”版本上正确运行。

9.2 “易模块”的使用方法



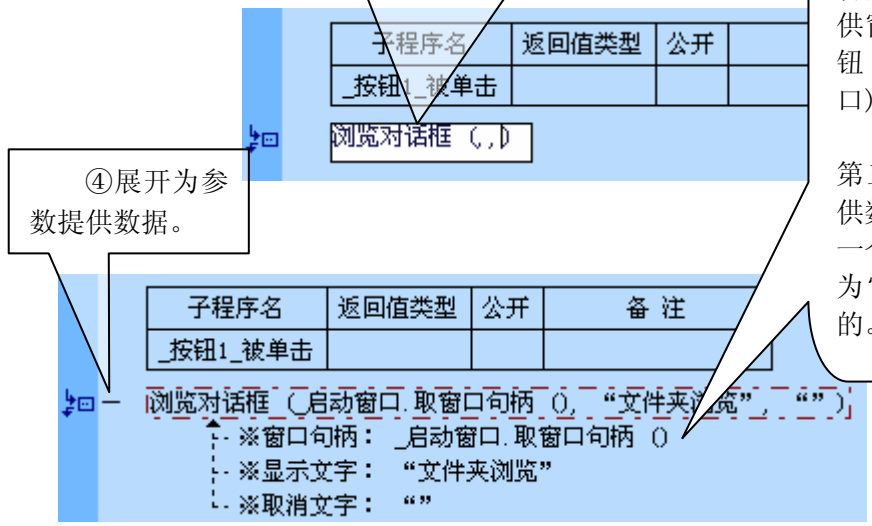


②粘贴到将被调用的其它子程序中。“Ctrl+V”键。

根据参数备注的要求，提供窗口句柄(按钮 1 的父窗口)。

如果不想给第三个参数提供数据，也要置一个空文本，因为它是文本型的。

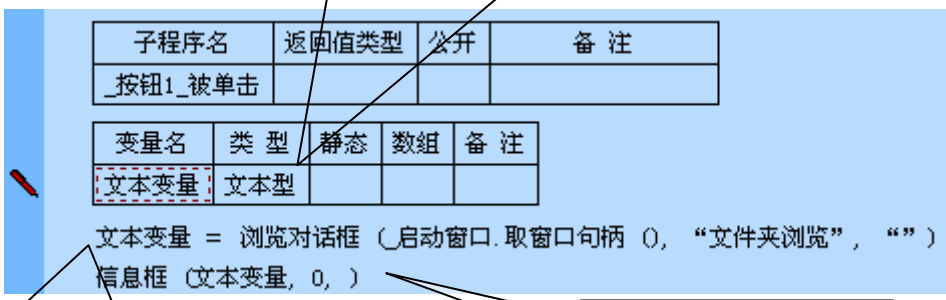
④展开为参数提供数据。



⑤按“F5”键，运行程序。单击“按钮 1”。

程序运行时，虽然可以选择文件夹，但单击“浏览文件夹”上的“确定”按钮看不到任何效果，所以还要用到这个模块设置的返回值。方法如下。

①在调用的子程序中添加文本型变量“文本变量”。



②既然有返回值，我们就可以把这个返回值保存到一个与返回值类型相同的变量中。

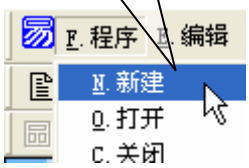
③添加信息框，显示返回值数据。

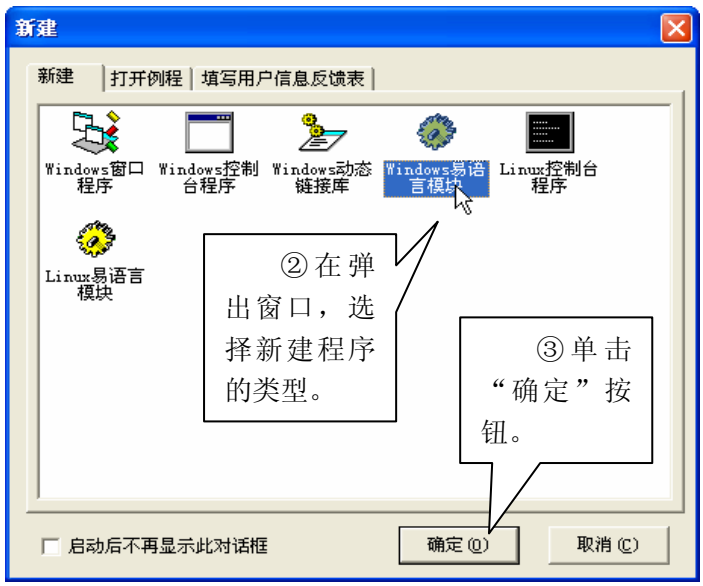
使用“易模块”的方法和使用“易语言”命令的方法一样；
也可以把“_接口程序集”里的子程序看作是自定义的子程序。

9.3 开始写第一个易模块

下面让大家练习从头定一个简单的易模块，它只有弹出一个信息框的功能。

① 通过“易语言”菜单，选择“新建”创建新程序。





进入代码编辑界面。

程序集名	备 注
程序集1	

子程序名	返回值类型	公开	备 注
_启动子程序	整数型		请在本子程序中放置易模块初始化代码

◇

_临时子程序 ()
※备注：在初始化代码执行完毕后调用测试代码

返回 (0)
※备注：可以根据您的需要返回任意数值

子程序名	返回值类型	公开	备 注
_临时子程序			

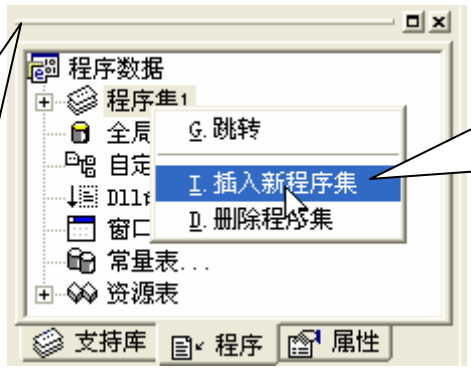
※备注：本名称子程序用作测试程序用，仅在开发及调试环境中有效，编译发布程序前将被系统自动清空，请将所有用作测试的临时代码放在本子程序中。 ***注意不要修改本子程序的名称、参数及返回值类型。

◇

可以在此处调用其它子程序（接口程序集中的子程序，后面将提到）。

在“易语言 3.5”以前版本中，程序为用户提供“_接口程序集”，将接口子程序写在这里。

在“易语言 3.6”中，大家可以直接勾选子程序的“公开”属性，也可以实现易声明模块的接口。



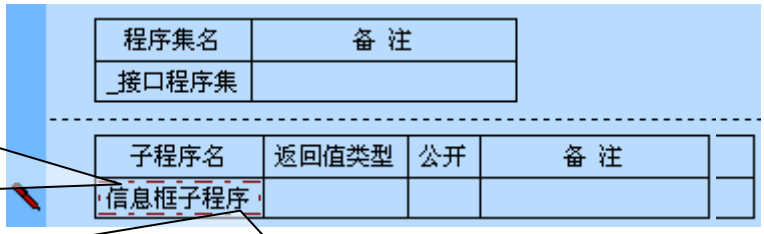
①点击鼠标右键，插入新程序集。

②将程序集名称改为“_接口程序集”。

程序集名	备 注
程序集2	

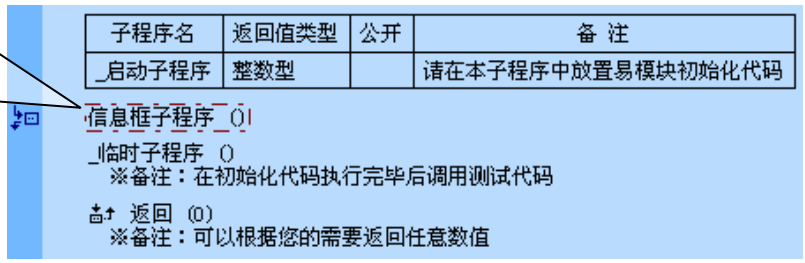
程序集名	备 注
接口程序集	

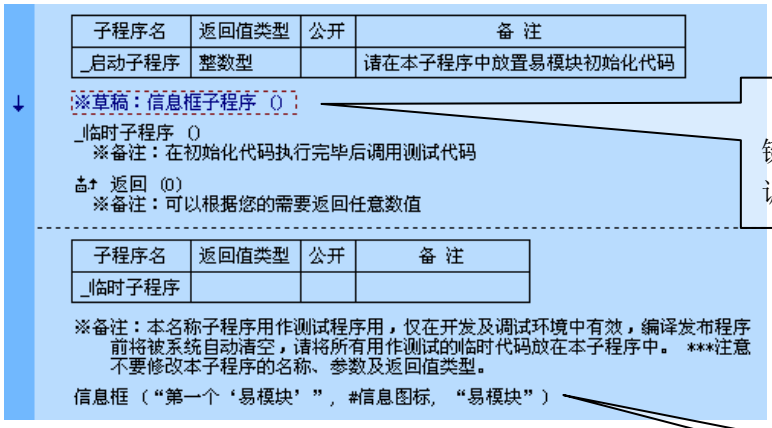
③新建子程序，并重命名为“信息框子程序”。



所有与此子程序要实现功能的代码都要添加到它的下面。而且编译后，所有代码被封装在模块中，使用者只可以看到子程序的结构，无法看到代码。

④ 把“信息框子程序”复制到“_启动子程序”。

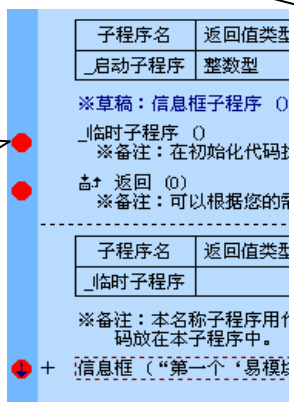




⑤按“Enter+Ctrl”键置为草稿。暂时不调用。

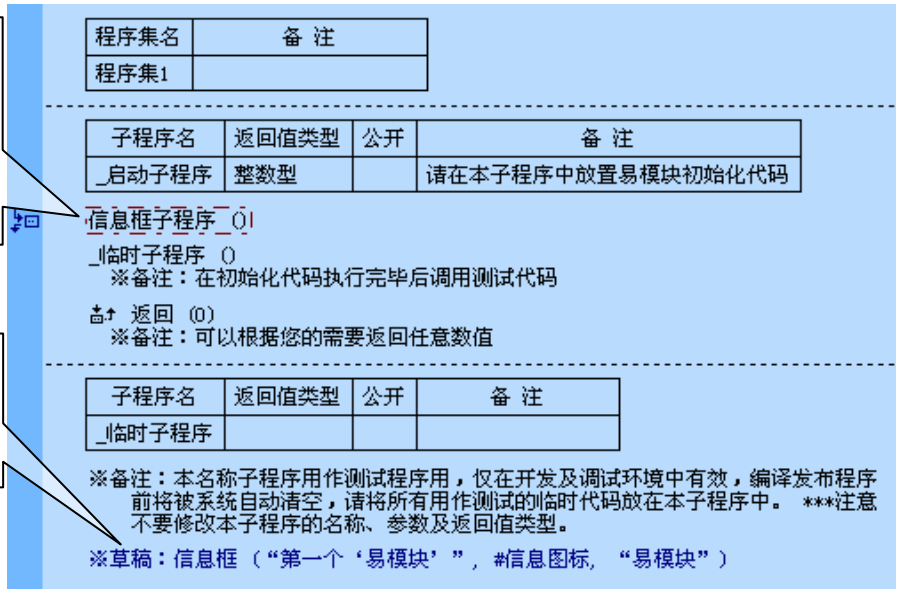
⑥添加信息框。

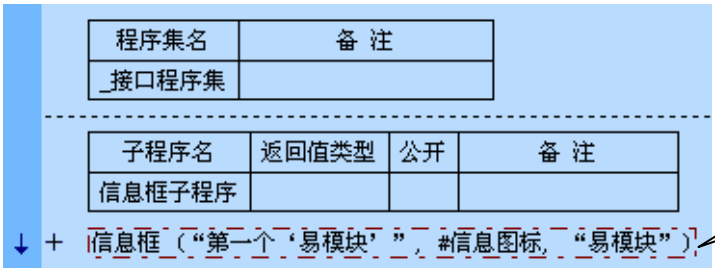
⑦设置断点，按“F5”键运行程序，跟踪程序的执行过程。



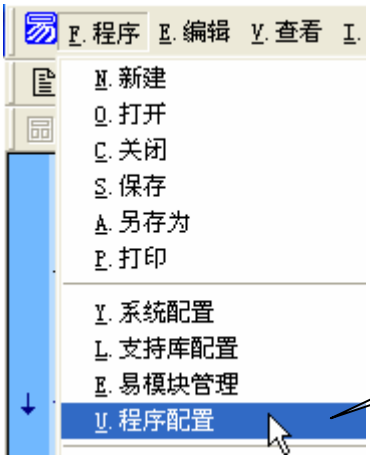
⑧将草稿重新置为代码，调用子程序。

⑨复制后置为草稿。



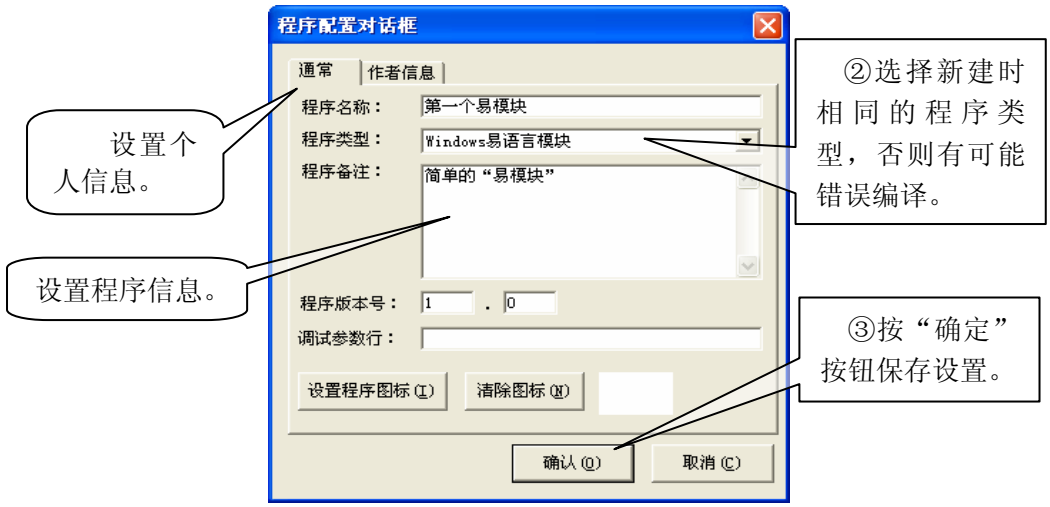


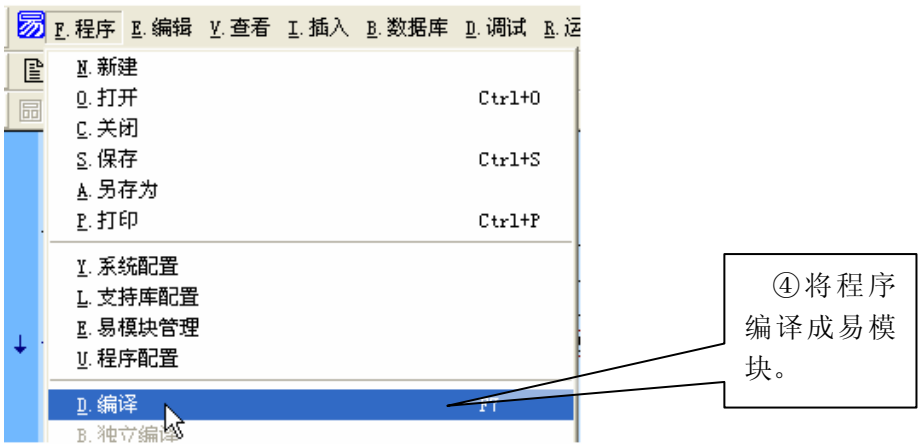
⑩粘贴到“_接口程序集”的“消息框子程序”中。按“F5”键运行程序。

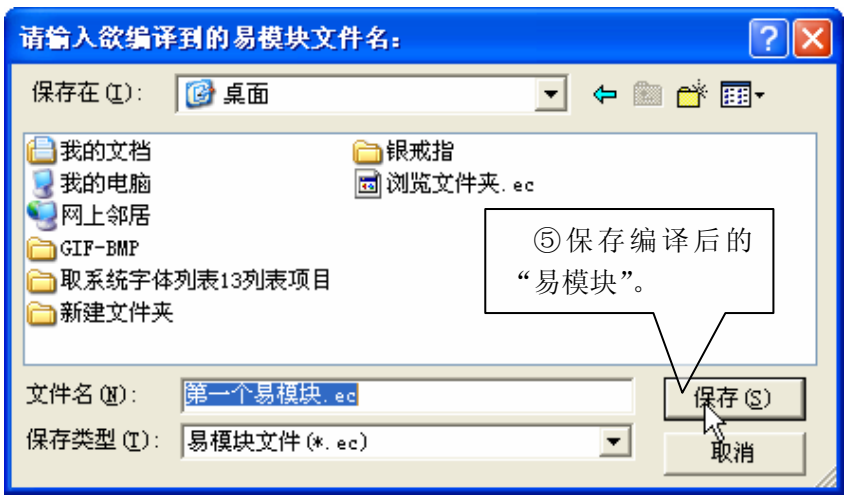


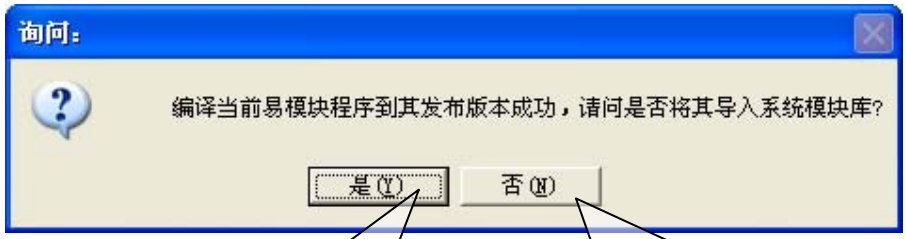
①设置程序信息。

一个简单的“易模块”创建完毕。下面开始编译并保存。


程序配置对话框
通常
作者信息
程序名称：
第一个易模块
程序类型：
Windows易语言模块
程序备注：
简单的“易模块”
程序版本号：
1
0
调试参数行：
设置程序图标(I)
清除图标(N)
确认(O)
取消(C)
设置个人信息。
设置程序信息。
②选择新建时相同的程序类型，否则有可能错误编译。
③按“确定”按钮保存设置。


F.程序
E.编辑
V.查看
I.插入
B.数据库
D.调试
R.运
N.新建
O.打开 Ctrl+O
C.关闭
S.保存 Ctrl+S
A.另存为
P.打印 Ctrl+P
Y.系统配置
L.支持库配置
E.易模块管理
U.程序配置
D.编译
B.独立编译
④将程序编译成易模块。


请输入欲编译到的易模块文件名：
保存在(I):
桌面
我的文档
我的电脑
网上邻居
GIF-BMP
取系统字体列表13列表项目
新建文件夹
银戒指
浏览文件夹.ec
⑤保存编译后的“易模块”。
文件名(N):
第一个易模块.ec
保存类型(T):
易模块文件(*.ec)
保存(S)
取消



⑥ 按“是”钮导入。

注意：
如果不导入，可以再以后通过“易模块管理”安装。前面提到过。

⑦ 生成后缀为“ec”的文件。

模块程序集创建完毕后，就可以直接使用该程序集中的子程序了，就如同这些子程序是自己所编写的一样。在编译易程序时所有被使用的易模块会自动被一起编译进去。

注意：

1．除了删除模块程序集中的不用子程序外，不要修改程序集中的任何地方，否则有可能会导致编译不通过；

2．如果想删除对某模块的使用，直接删除为该模块所建立的模块程序集即可。

9.4 课后练习

1．安装和使用本章所创建的易模块。

2、将求两数和与返回最大数的子程序改为易模块使用。

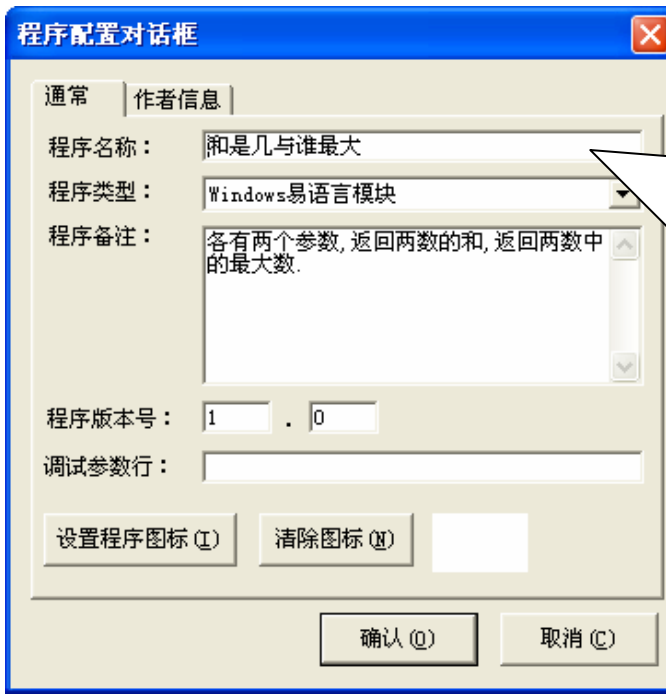
子程序名	返回值类型	公开	备 注		
和是几	整数型	✔			
参数名	类 型	参考	可空	数组	备 注
加数1	整数型				
加数2	整数型				

返回 (加数1 + 加数2)

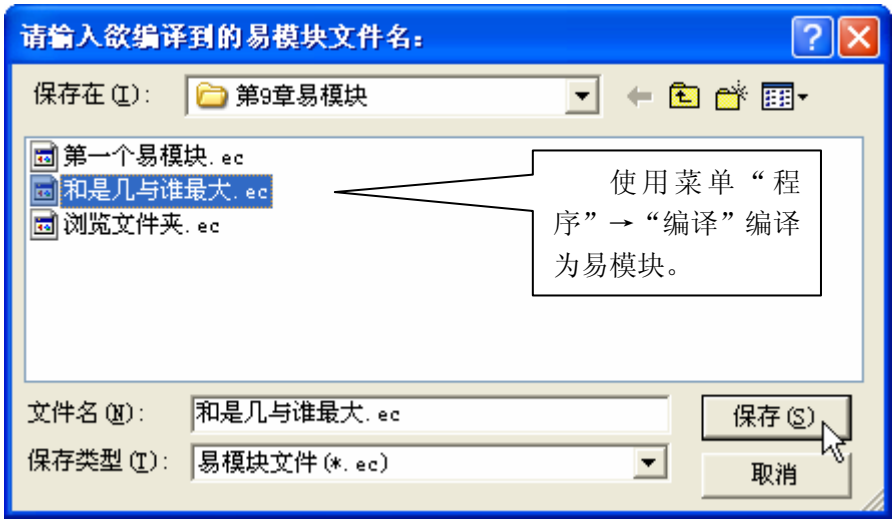
子程序名	返回值类型	公开	备 注		
谁最大	整数型	✔			
参数名	类 型	参考	可空	数组	备 注
整数1	整数型				
整数2	整数型				

返回 (选择 (整数1 > 整数2, 整数1, 整数2))

新建一个易模块，自动产生的“_启动子程序”和“_临时子程序”都不要管，再新建两个子程序，各有两个整数型参数，公开属性上要打勾，设置返回整数型值。再输入返回命令。

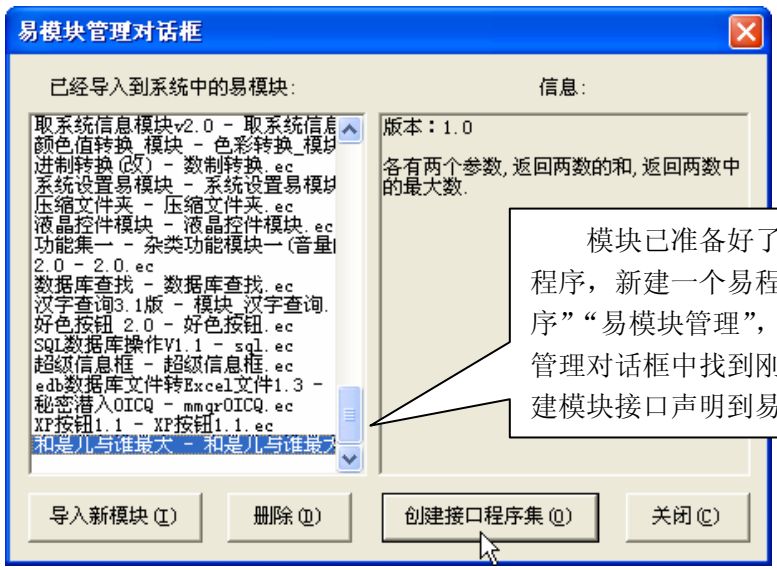


点击菜单“程序”→“程序配置”，可以调出程序配置对话框。写程序名称及程序备注，以备以后或提供给其他人调用时参看。

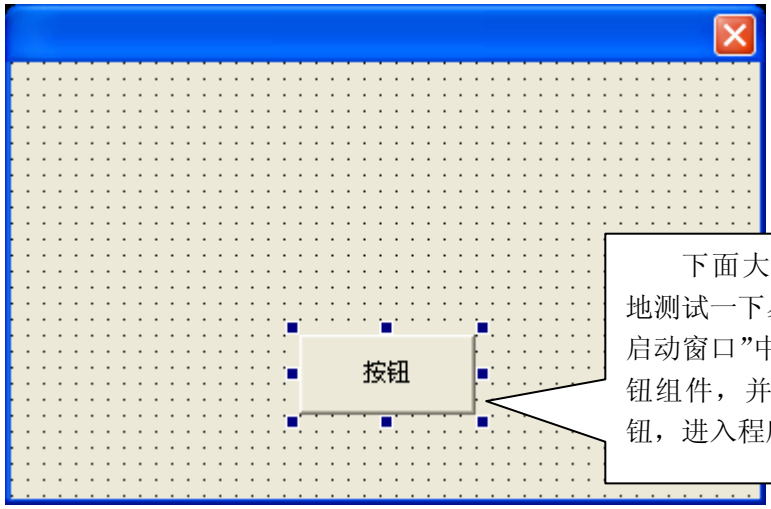


使用菜单“程序”→“编译”编译为易模块。

编译为易模块完成后提示是否导入系统模块库，请点“是”钮。



模块已准备好了，请关闭易模块程序，新建一个易程序，点菜单“程序”“易模块管理”，在弹出的易模块管理对话框中找到刚才的易模块，创建模块接口声明到易程序中。



下面大家可以简单地测试一下易模块，在“_启动窗口”中加入一个按钮组件，并双击这个按钮，进入程序设计界面。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

```
信息框 (和是几 (3, 4), 0, )
信息框 (谁最大 (3, 8), 0, )
```

在“_按钮 1_被单击”子程序中输入以下两行程序代码：

```
信息框 (和是几 (3, 4), 0, )
信息框 (谁最大 (3, 8), 0, )
```

最后按 F5 键试运行，点击按钮就可以看到结果。

第 10 章　API 函数的应用

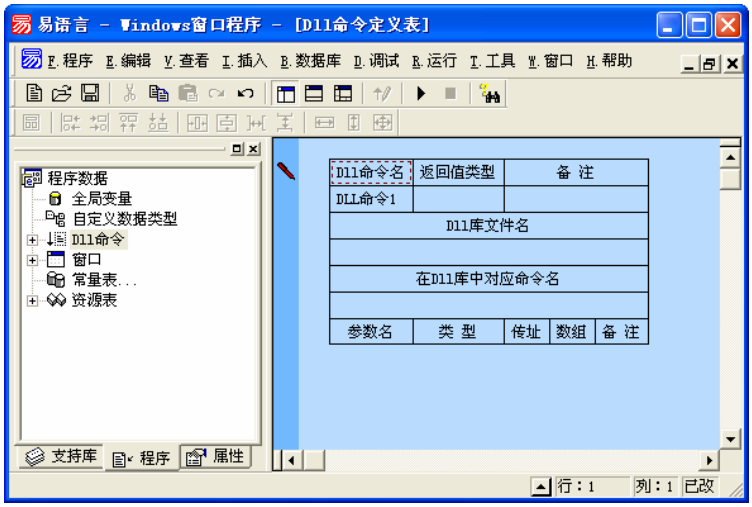
API 函数，也称 DLL 命令，是 Windows 系统外部动态连接库（即 DLL 库）中的命令。和 VB,VC 一样，易语言对 API 也有很好的支持。API 是 Windows 的基础，学会使用 API 就可以实现 Windows 绝大部分的功能。

本章学习内容：

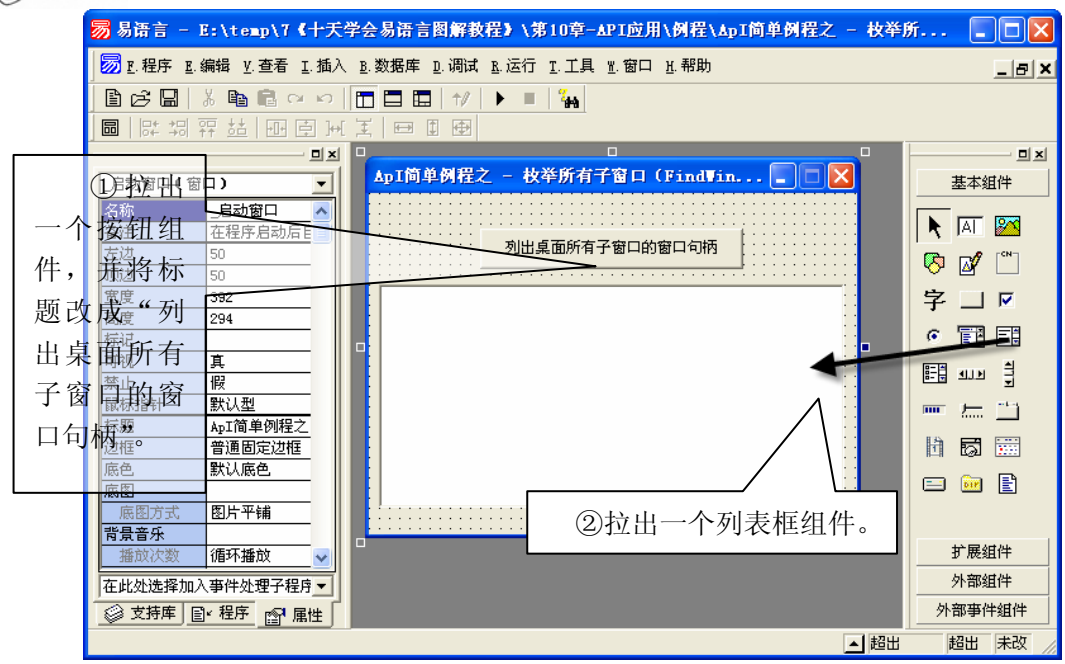
10.1 如何定义 API 函数

易语言中，使用一个API函数前（也称DLL命令），首先要对该函数进行定义，定义DLL命令涉及到以下主要属性：

Dll命令名、返回值类型、Dll库文件名、Dll命令在Dll库中的对应命令名、Dll命令参数。



下面用一个可以取出当前所有的窗口句柄的例程，来进一步了解DLL命令的定义方法和使用。



注意：API 资料如何查找？

Windos 中的 DLL 命令有很多，网上也有许多 API 的帮助文件，列出了常用的 API 令的相关资料。使用 API 之前，可以先上网下载一个 API 的帮助文档。

下面会用到 1 个 DLL 命令：FindWindowExA，这个命令可以在窗口列表中寻找与指定条件相符的第一个子窗口。

④新建一个 DLL 命令。

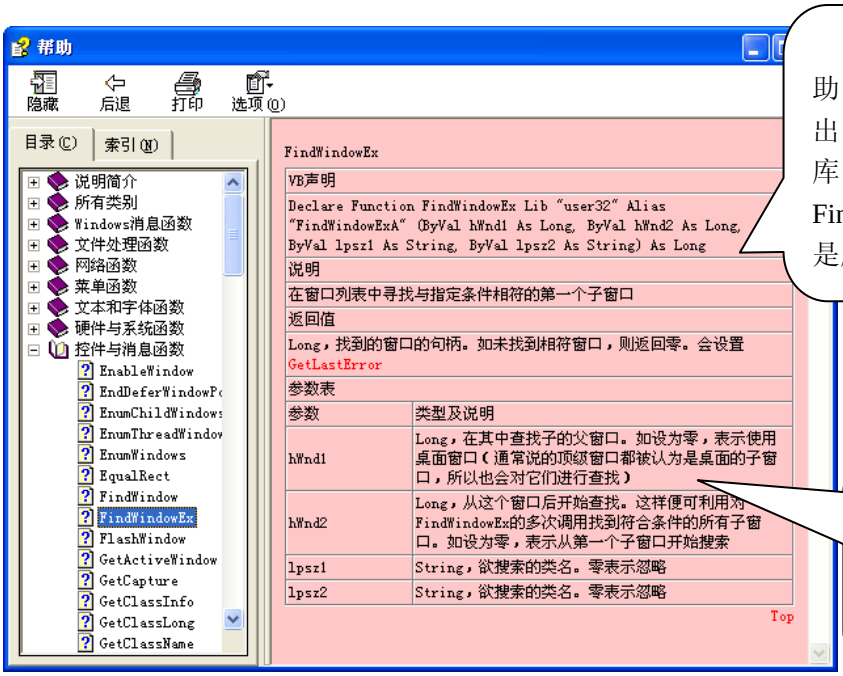
Dll命令名	返回值类型	备 注		
取子窗口句柄	整数型			
Dll库文件名				
user32				
在Dll库中对应命令名				
FindWindowExA				
参数名	类 型	传址	数组	备 注
hWnd1	整数型			
hWnd2	整数型			
lpsz1	整数型			
lpsz2	整数型			

定义 DLL 命令在易语言中使用的命令名，可以随意定义，最好是定义一个方便识别的命令名

定义 DLL 库文件名。按照 API 资料中填写，不可以自定义。

定义 DLL 库中对应的命令名。按照 API 资料中填写，不可以自定义。

API 资料中描述的，有几个参数，就添加几个参数，并且参数的类型要和资料中的符合，参数名可以自定义。



从 API 的帮助文件里可以查出 User32 是 DLL 库文件名，FindWindowExA 是库命令名。

这里有四个参数的类型与说明。

⑥双击按钮，在_按钮 1_被单击的子程序下输入代码：

```
列表框 1.清空 ()
hWnd = 取子窗口句柄 (0, 0, 0, 0)
判断循环首 (hWnd ≠ 0)
    列表框 1.加入项目 (到文本 (hWnd), )
    hWnd = 取子窗口句柄 (0, hWnd, 0, 0)
判断循环尾 ()
```

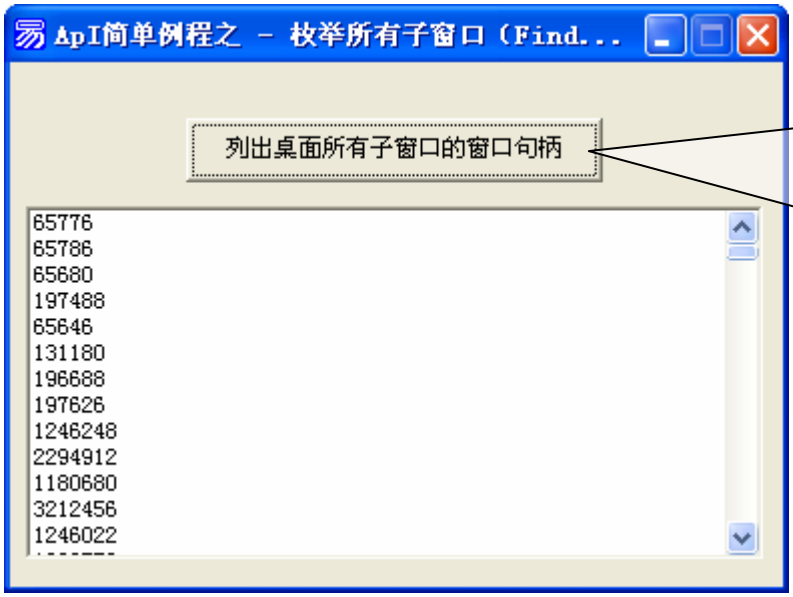
⑤新建一个整数型的变量。

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

变量名	类 型	静态	数组	备 注
hWnd	整数型			

```
列表框1.清空 ()
hWnd = 取子窗口句柄 (0, 0, 0, 0)
判断循环首 (hWnd ≠ 0)
    列表框1.加入项目 (到文本 (hWnd), )
    hWnd = 取子窗口句柄 (0, hWnd, 0, 0)
判断循环尾 ()
```

用判断循环首命令，循环取出当前所有子窗口的窗口句柄。



按下 F5 键试运行，点击按钮后，列表框就列出了所有子窗口的窗口句柄了。

10.2 API 函数的综合应用

用 API 函数，可以实现很多特殊的效果，下面练习用 3 个 DLL 命令实现窗体的透明效果。

用到了 3 个 DLL 命令，分别是：

GetWindowLong：从指定窗口的结构中取得信息。

SetWindowLong：在窗口结构中为指定的窗口设置信息。

SetLayeredWindowAttributes：设置窗体的透明色。

Dll命令名	返回值类型	备 注
SetWindowLong	整数型	
Dll库文件名		
在Dll库中对应命令名		
SetWindowLongA		
参数名	类 型	传址	数组	备 注
hwnd	整数型			
nIndex	整数型			
dwNewLong	整数型			

③对这 3 个 DLL 命令进行定义。

Dll命令名	返回值类型	备 注
SetLayeredWindowAttributes	整数型	
Dll库文件名		
在Dll库中对应命令名		
SetLayeredWindowAttributes		
参数名	类 型	传址	数组	备 注
hwnd	整数型			
crKey	整数型			
bAlpha	整数型			
dwFlags	整数型			

Dll命令名	返回值类型	备 注
GetWindowLong	整数型	
Dll库文件名		
在Dll库中对应命令名		
GetWindowLongA		
参数名	类 型	传址	数组	备 注
hwnd	整数型			
nIndex	整数型			

如果库文件名是：Kernel32.dll、Gdi32.dll、User32.dll 、 Mpr.dll 、 Advapi32.dll 填写时可以省略，系统会自动搜索。

注意：API 函数中常量的使用

使用 DLL 命令时，会有一些参数用到了 API 中的常量值，这些常量值可以使用 API 常量查询工具查找，并在易语言中使用自定义常量，进行定义，也可以直接将常量值添入参数中。

④双击_启动窗口，产生_启动窗口_创建完毕子程序，然后新建一个整数型的变量“Ret”

子程序名	返回值类型	公开	备 注
_启动窗口_创建完毕			

变量名	类 型	静态	数组	备 注
Ret	整数型			

```
Ret = GetWindowLong (取窗口句柄 (), -20)
Ret = 位或 (Ret, 524288)
SetWindowLong (取窗口句柄 (), -20, Ret)
SetLayeredWindowAttributes (_启动窗口.取窗口句柄 (), 0, 255, #LWA_ALPHA)
```

⑤在子程序下输入代码：

```
Ret = GetWindowLong (取窗口句柄 (), -20)
Ret = 位或 (Ret, 524288)
SetWindowLong (取窗口句柄 (), -20, Ret)
SetLayeredWindowAttributes (_启动窗口.取窗口句柄 (), 0, 255, #LWA_ALPHA)
```

⑥在滑块条的事件列表中选择“位置被改变”事件，产生“_滑块条 1_位置被改变”子程序。

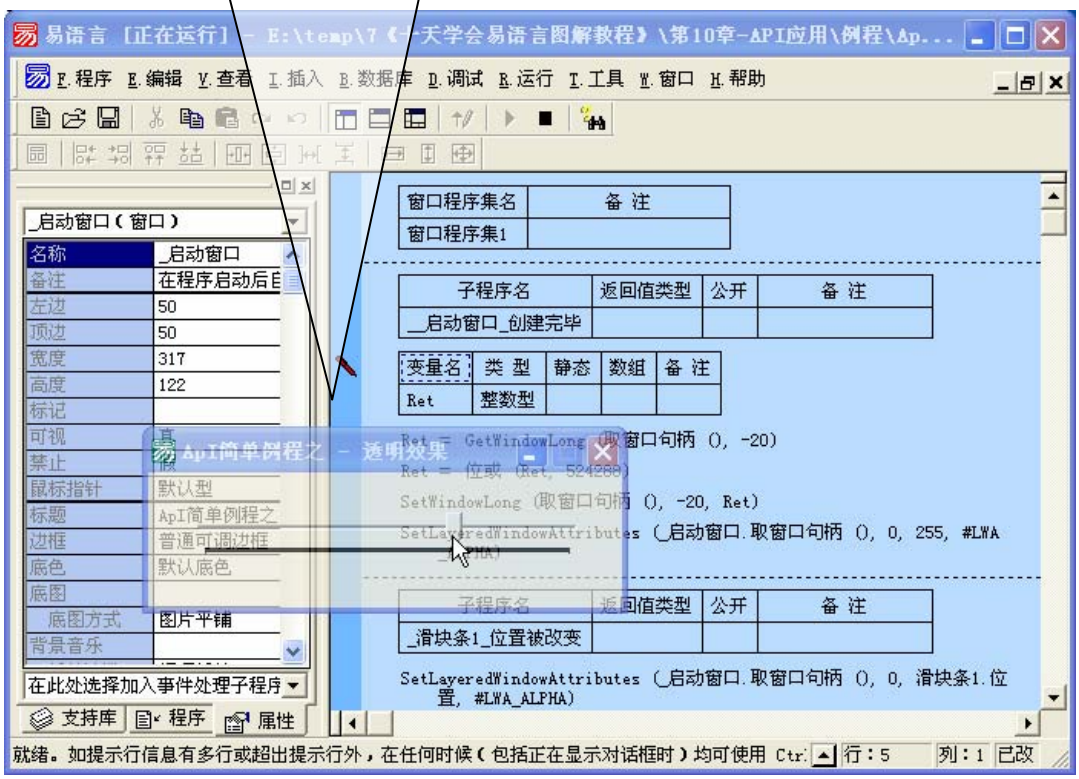
子程序名	返回值类型	公开	备 注
_滑块条1_位置被改变			

```
SetLayeredWindowAttributes (_启动窗口.取窗口句柄 (), 0, 滑块条1.位置, #LWA_ALPHA)
```

⑦在子程序下输入代码：

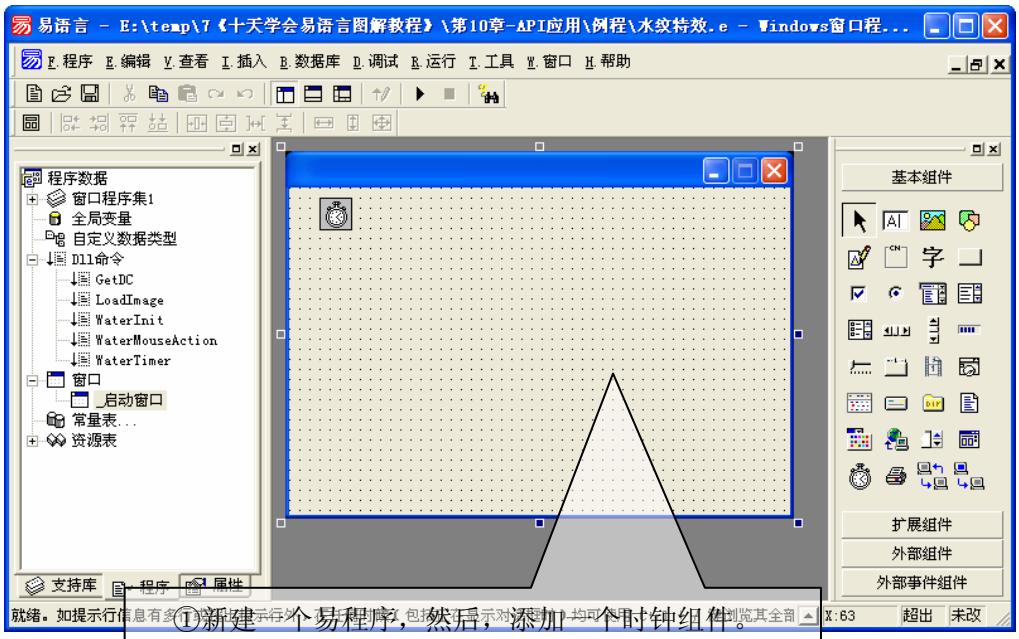
```
SetLayeredWindowAttributes (_启动窗口.取窗口句柄 (), 0, 滑块条 1.位置, #LWA_ALPHA)
```

运行程序，然后拖动滑块条，可以看到，窗体渐渐变的透明，直到完全透明。



10.3 外部 DLL 库的应用

前 2 节都是介绍基本 Windows API 库文件，但 API 的使用方法多种多样，不但可以使用 Windows 自带的 DLL 命令，而且可以使用外部的 DLL 库中的 DLL 命令，本节就使用一个外部的 DLL 库，来做一个水纹特效。



①新建一个易程序，然后，添加一个时钟组件。

注意：

该例程要使用到一个外部的 DLL 库，所以，在编写程序前，先从随书光盘中将 WaterDll.dll 文件拷贝到本地机器上，并一同将随书光盘中的“例图 01.bmp”拷贝到程序保存到的目录下。

本例程要用到 2 个默认 DLL 库中的命令，和 3 个外部 DLL 库中的命令。

2 个 user32.dll 中的命令：

GetDC：获取指定窗口的设备场景

LoadImage：载入一个位图、图标或指针

3 个 WaterDll.dll 中的命令（外部 DLL 库）：

外部 DLL 库中的命令使用方法，参见外部 DLL 库自带的帮助文档，没有帮助文档也可参见例程，或在实际使用的时候给命令参数不同的值看看出现的效果，自己理解 DLL 库中命令的作用。

Dll命令名	返回值类型	备 注		
GetDC	整数型			
Dll库文件名				
在Dll库中对应命令名				
GetDC				
参数名	类 型	传址	数组	备 注
hwnd	整数型			

Dll命令名	返回值类型	备 注		
LoadImage	整数型			
Dll库文件名				
在Dll库中对应命令名				
LoadImageA				
参数名	类 型	传址	数组	备 注
hInst	整数型			
lpsz	文本型			
un1	整数型			
n1	整数型			
n2	整数型			
un2	整数型			

② 定义默认 DLL 库中的命令。

Dll命令名	返回值类型	备 注
WaterInit	整数型	
Dll库文件名		
waterdll.dll		
在Dll库中对应命令名		
WaterInit		

参数名	类 型	传址	数组	备 注
bitmap	整数型			

③ 定义外部 DLL 库中的命令。

由于是外部 DLL 库，所以，DLL 库文件名要填写清楚。

Dll命令名	返回值类型	备 注
WaterMouseAction	整数型	
Dll库文件名		
waterdll.dll		
在Dll库中对应命令名		
WaterMouseAction		

参数名	类 型	传址	数组	备 注
hdc	整数型			
sx	整数型			
sy	整数型			
mx	整数型			
my	整数型			
half	整数型			扩展半径
deep	整数型			扩展深度

命令中各参数可以照书中填写，各参数的作用，可以编写完程序后，更改参数的值，进行调试，就可以了解命令中的各个参数了。

Dll命令名	返回值类型	备 注
WaterTimer	整数型	
Dll库文件名		
waterdll.dll		
在Dll库中对应命令名		
WaterTimer		

参数名	类 型	传址	数组	备 注
hdc	整数型			
sx	整数型			
sy	整数型			

④双击_启动窗口产生了“_启动窗口_创建完毕“的子程序，并新建一个整数型的变量“bitmap”。用来存放 LoadImage 命令的返回值。

子程序名	返回值类型	公开	备 注
_启动窗口_创建完毕			

变量名	类 型	静态	数组	备 注
bitmap	整数型			

```
时钟1.时钟周期 = 10
bitmap = LoadImage (0, 取运行目录 () + “\例图01.bmp”, 0, 0, 0, 16)
WaterInit (bitmap)
```

16是#LR_LOADREALSIZE的常量值，使用常量查询工具查出。

⑤在_启动窗口_创建完毕的子程序中输入代码：

```
时钟1.时钟周期 = 10
bitmap = LoadImage (0, 取运行目录 () + “\例图01.bmp”, 0, 0, 0,
#LR_LOADFROMFILE)
WaterInit (bitmap)
```

子程序名	返回值类型	公开	备 注
_时钟1_周期事件			

```
WaterTimer (GetDC (_启动窗口.取窗口句柄 ()), 0, 0)
```

⑥双击时钟组件，产生“_时钟1_周期事件”子程序，子程序下输入代码：

```
WaterTimer (GetDC (_启动窗口.取窗口句柄 ()), 0, 0)
```

子程序名	返回值类型	公开	备 注		
_启动窗口_鼠标位置被移动	逻辑型				
参数名	类 型	参考	可空	数组	备 注
横向位置	整数型				
纵向位置	整数型				
功能键状态	整数型				

```
WaterMouseAction (GetDC (_启动窗口.取窗口句柄 ()), 0, 0, 横向位置, 纵向位置, 5, 100)
```

⑦将子程序的返回值设置成“逻辑型”。

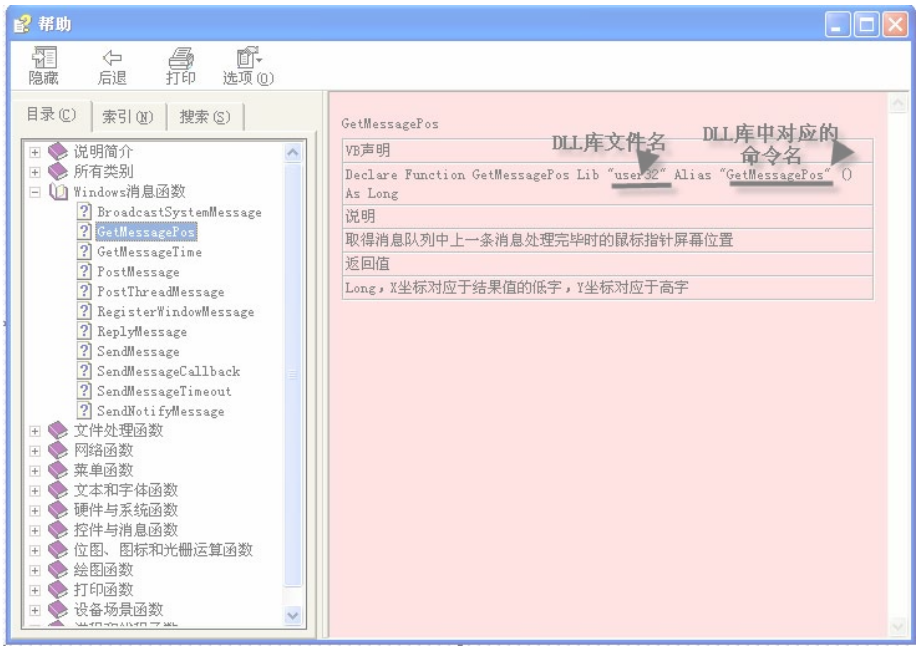
⑧在启动窗口的事件选择框中选择“鼠标位置被移动”选项，产生了“_启动窗口_鼠标位置被移动”子程序，然后在子程序下输入代码：

```
WaterMouseAction (GetDC (_启动窗口.取窗口句柄 ()), 0, 0, 横向位置, 纵向位置, 5, 100)
```

试运行程序，然后在窗体上移动鼠标，就可以看到逼真的水纹效果了。

10.4 课后练习

（1）学着使用 API 函数的帮助文档：



(2)使用简单的 API，制作一个模拟键盘按下“Win 键+D”，和模拟鼠标移动的程序。

Dll命令名	返回值类型	备 注
模拟键盘		
Dll库文件名		
在Dll库中对应命令名		
keybd_event		
参数名	类 型	传址	数组	备 注
欲模拟的虚拟键码	整数型			
键的OEM扫描码	整数型			
常量	整数型			
参数	整数型			

Dll命令名	返回值类型	备 注
模拟鼠标移动	整数型	
Dll库文件名		
在Dll库中对应命令名		
SetCursorPos		
参数名	类 型	传址	数组	备 注
x	整数型			
y	整数型			

子程序名	返回值类型	公开	备 注
_按钮1_被单击			

```
模拟键盘 (#VK_LWIN, 0, 0, 0)
模拟键盘 (#D键, 0, 0, 0)
模拟键盘 (#VK_LWIN, 0, #KEYEVENTF_KEYUP, 0)
※备注：其实就是模拟按下WIN键+D键（或WIN键+M键）
※备注：注意按下后要放开键
```

子程序名	返回值类型	公开	备 注
_按钮2_被单击			

```
模拟鼠标移动 (0, 0)
```

提示：GetMessagePos：模拟鼠标移动；keybd_event：模拟键盘按下。

这里是 DLL 声明，分别声明模拟键盘与模拟鼠标移动的两个 DLL 命令。

在_启动窗口中新建两个按钮，分别双击按钮组件，以进入被单击事件子程序，并输入程序代码。

最后按 F5 键试运行，看看效果。